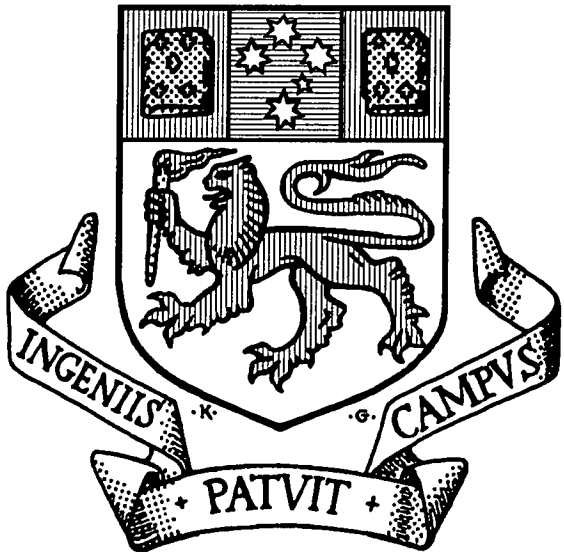
Hardware and Software Development of the Anglescan Tracking System



The University of Tasmania

Hardware and Software Development of the Anglescan Tracking System

By

Vito Dirita (BE Electronic Engineering)

Submitted in fulfilment of the requirements for the degree of

MASTER OF ENGINEERING SCIENCE

The University Of Tasmania
Department of Electrical and Electronic Engineering
Hobart Tasmania. 1994

Supervisor: Mr. G. The'
Submission Date: January 1995

DECLARATION:

I declare that this thesis is original except where due acknowledgement is given, and the material has not been accepted for the award of any other degree or diploma.

Vito. Dirita

INDEX

Chapter 1: Introduction:

Chapter 2: Hardware/Software Design:

Chapter 3: System Identification:

Model Identification Applied

Measurements and Model Validation

Chapter 4: Controller Design:

Introduction

Compensator Design

Chapter 5: Instrument Calibration:

Mathematical Background

Simulation/ Results

Chapter 6: Discussion/Conclusion:

Chapter 7: Appendix:

Mathematical Symbols Table

References

Mathematical Derivations

Simulation Programs:

Circuit Diagrams

Aglescan Tracking Software

Photographs

Abstract:

In this thesis, the author describes all the developmental work of an automated surveying tracking theodolite referred to as *The Anglescan Tracking System.* This system consists of an optical system, an electronic system (with detailed schematics and printed circuit artwork, included) the software system which is approximately 100 pages of C source code, system identification and controller simulation, modelling and design, and finally instrument calibration and testing effort over a 2 year period.
The project is a collaboration between the Department of Electrical and Electronic Engineering and the School of Surveying at the University of Tasmania. It is supported by an ARC university research fund granted to the School of Surveying over a two year period.

The initial investigation into the feasibility of the project was carried out by Dr. A. Sprent, then at the North East London Polytechnic (Sprent, 1.12). During this time the optical system of the instrument was developed. Following this investigation, it was decided to further undertake the continuation of the project as a Master's Degree in order to develop the electronics, measurement, tracking and control software.

The Anglescan system is similar to a standard surveying theodolite mounted on a tripod and capable of determining the position of a retro-reflective mirror based within the instrument's field of view with respect with its optical centre of field (in both azimuth and elevation planes). This involves a new and different technique for target location than the currently existing electronic theodolites (see sections 1.5, 1.6).

One of the main advantages of this method is the improved target locating accuracy, it has been field tested with an accuracy of +/-1 second of arc, and simplicity in optical design. The disadvantages however include increased complexity of both the hardware (electronics), software, and calibration involved.
The instrument deviates from a standard theodolite in that it has been fitted with precision DC servomotor drives on the azimuth and elevation axes for target tracking. The servomotors also include optical rotary encoders for measurement..

While tracking, the system operates by continuously locating the target mirror with respect to its centre of field of view, then applying a feedback control signal to the azimuth and elevation motors in order to re-centre the instrument directly onto the target, this is updated at a rate of 30 samples/second.

Part of the objectives of this work is to design the necessary hardware and software to achieve this task, this includes the design of a return (laser) analog signal processor card, the design of a target position measurement card, and azimuth/elevation servo motors interface cards (azimuth card and elevation card).

Furthermore, the project also requires the design of an appropriate target acquisition and control loop strategy to allow smooth and accurate tracking of a moving target. A number of control system designs have been investigated, including proportional control, pole placement design and linear quadratic control. We have also used system identification techniques to obtain an accurate model of the plant which is necessary for controller design and simulation.

To achieve the high precision required in measurement, it is necessary to calibrate the instrument. The calibration method is a complex procedure which uses least squares techniques to produce the required calibration accuracy (this is discussed in detail in section 5).

The instrument finds applications in areas such as monitoring earth movements, motion of large structures such as building and bridges, locating earth working machinery such as tractors, bulldozers, graders, concrete and asphalt paving machines where an accurate level surface is required.
Another application of particular interest is in surface contour mapping where we require to map a 3 dimensional surface (obtain the surface contour or topography), this application is a slightly more complex problem requiring the use of two instruments, this then becomes a multivariable control problem.

As a result of this research work, one paper was presented at the **IREECON-91** conference in Sydney and is published in the proceedings [Dirita 1.13]. Another paper was presented at the **CONTROL-92** conference held in Perth in November 1992 and is published in the proceedings [Dirita 4.4].

Acknowledgements:

The author is grateful to many of the members of academic staff both in the School of Surveying and Department of Electrical and Electronic Engineering, whose assistance has contributed greatly to the preparation of this thesis.

In particular, my supervisor Mr. G. The' for his constant guidance and encouragement during the course of the project.
I would also like to thank my second supervisor Dr. A. Sprent for providing both the research project and financial assistance throught the ARC grant.

Finally, my thanks to Mr. K. Bolton in the Physics department, for the preparation of printed circuit artwork photonegatives from the Protel AUTOTRAX gerber files, and etching the printed circuit boards.

V. Dirita
The University of Tasmania
.....
HOBART TASMANIA

Thesis Outline:

The thesis covers a broad aspect of engineering disciplines such as embedded software design, hardware design, control system design and simulation, system identification, instrument calibration, and testing. Each has been arranged into individual chapters as outlined below:

Chapter 1: Introduction Starts with and introduction to the electronic theodolite and surveying instrumentation, it also describes the principles of some existing automatic optical tracking systems. Then an introduction to the principle of operation of the anglescan system is given.

Chapter 2: Hardware and Software Design Covers a detailed overview of the principle of operation, and an introduction to the optical system, including the measurement technique and potential sources of errors. This chapter is grouped into 3 parts: The first is a description of the optical system, the second section is a detailed description of the entire electronic system which is based on a dedicated bus backplane and low end Z80 microprocessor (due to be upgraded to a full 32 bit off-the-shelf processor system). The third section of this chapter is a functional decomposition and top-down design of the Anglescan tracking software system totally developed in C high level language, it also describes the user menu interface and command structure for operating the instrument.

Chapter 3: System Identification This chapter is devoted to system identification. It describes the most common techniques of identification, we also develop the structure of the servosystem model, then proceed to apply least squares techniques for estimating model coefficients using a pseudo random binary excitation source. Several models are compared both higher order and reduced order form. The model are then verified by step response simulation and compared with the real system response.

Chapter 4: Controller Design Covers the subject of controller design. We commence with the transient behaviour and steady state performance of the uncompensated system. Various forms of controller compensation are subsequently applied, these include the most common implementations of Proportional control and Pole placement design. These models are verified by computer simulation and compared with the response from the real system. We also describe the measurement model which is included in the control system model.

Chapter 5: Instrument Calibration Covers the topic on instrument calibration. Instrument calibration requires the derivation of a system of equations describing the measurement data as a function of calibration unknowns (which turn out to be non linear), applying linearization techniques and least squares for parameter estimation. This is verified by computer simulation.

Chapter 6: Discussion and Conclusion This chapter covers typical applications of the anglescan system, including surface topography determination, tracking of moving targets and general survey use. We also discuss possible areas for future development. A discussion and conclusion is also included.

Chapters 7: Appendix Contains a table of mathematical symbols, general notations used, references for further reading, a listing of the simulation programs used, schematic diagrams for each board, some photographs of the system, and information on the user menu interface. The anglescan software system is too long and will not be reproduced in the thesis.

General Notation and Terminology:

Scalars:	Scalars are denoted in lower case plain text, ex: x
Vectors:	Vectors are denoted in lower case plain underlined text, ex: $\underline{\theta}$
Matrices:	Matrices are denoted in upper case bold text, eg: **H**
Subscripts:	- Iteration number, eg: θ_k= k'th iteration - Variables belonging to same subset, eg: $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$ - Sampling: eg: x_k=k'th sampled value of x(t) at t=k.T
Constants:	Denoted in upper case plain text, ex: K
Estimates:	Is the original variable annotated by a hat symbol: eg: $\hat{x}$
Transformed values:	Are represented in upper case thus H(s) and H(z) are the laplace transform and the Z transform.of h(t) respectively.
Time derivatives:	Represented in dot notation: eg $\dot{e}$ = de/dt
Shift Operator:	Denoted by q^{-1} is the backward shift operator.

Chapter 1

Introduction

[1.1] Electronic Theodolites:

Here we discuss the various components and functions of a basic electronic theodolite prior to describing the Anglescan Tracking System. Refer to Kennie [1.6], Deumlich [1.8], and Laurilla [1.9] for further references on theodolite contruction. Theodolites traditionally have been made for two different applications:

[a] Low accuracy instruments which are gnerally simpler, used for simple technical measurements such as on construction sites. Accuracy of the order of 1/10 arc minute is generally sufficient.

[b] Instruments of medium to high precision are used for triangulation, traversing and levelling , and for engineering surveys. Precision instruments are required for astro-geodetic measurements like azimuth, latitude, for triangulation and first or second order levelling.

A theodolite is generally required to perform the following functions:

(a) Levelling and alignment.
(b) Angular azimuth and elevation measurements.
(c) Distance measurement (DME).

Theodolite Construction:

The theodolite consists of the following main parts: referring to Fig.1.1, a fixed base with tribrach, a movable upper part and telescope. The base, with the tribrach, is attached to the tripod head and is levelled by adjusting three foot screws. On the upper part (the alidade), which is rotatable about the vertical axis and also contains the horizontal reading circle and a spirit level (circular or cylindrical). The two standards for the tilting (horizontal) axis are fixed, bearing the telescope with the sighting axis and vertical circle. The vertical circle reads zero when the telescope is pointing to the vertical (zenith) and 90 degrees when sighting the horizon. The horizontal circle is resettable in any direction .

For approximate levelling of the base, a circular (bull's eye) spirit level is used to provide levelling in 2 planes directly; for more accurate levelling however, a tubular level is used which provides levelling in one plane only, thus to level in both planes with the tubular level requires that the instrument is levelled in one plane, then rotated 90 degrees about the vertical axis and levelled again.

The instrument is centred over the station point by means of a plumb-bob or a built in optical plummet, and trivet stage (or stage plate) carrying the theodolite. The trivet stage can be moved laterally on the tripod head until the plumb-bob lies over the ground mark. Fig 1.1 below illustrates the components of a basic theodolite.

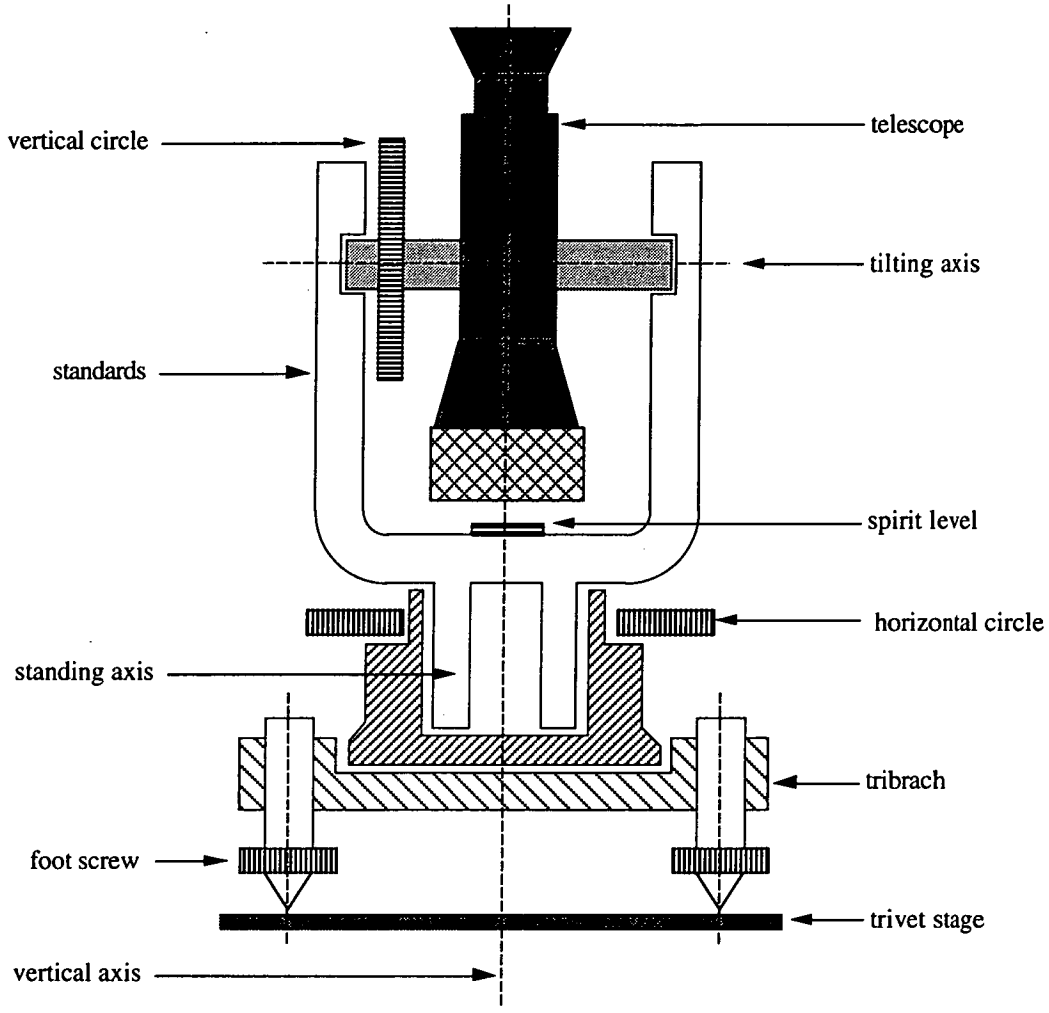


Fig.1.1 Basic theodolite construction

(a) Levelling and Alignment

Traditionally, consisting of spirit levels or tubular levels. Currently however, electronic levels utilize inverted pendulum techniques, and laser collimation techniques to achieve higher accuracy are being used, fig.1.2a below shows the construction of a standard Tubular spirit level, consisting of a cylindrical glass tube filled with a low viscosity liquid such as alcohol or ether.

Fig 1.2b below also shows the construction of a circular (Bull's eye) level, allows alignment in both planes, sensitivity is generally not better than 8 seconds of arc, used for quick-approximate levelling of a plane.

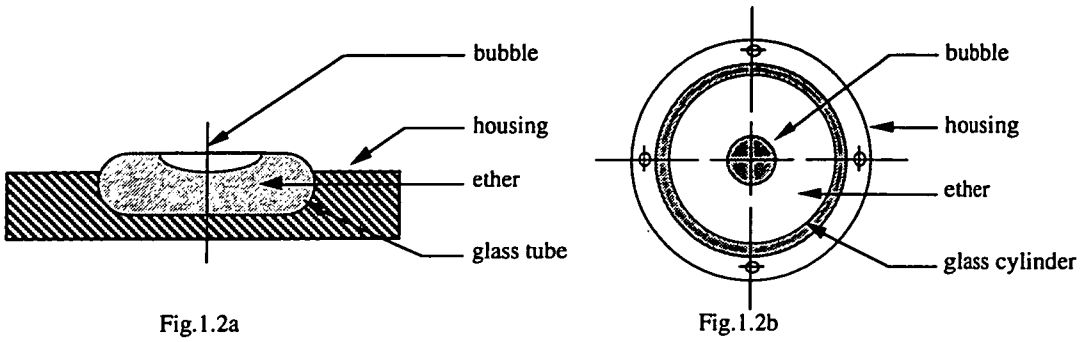


Fig.1.2a Fig.1.2b

Fig.1.2 Tubular and bull's eye spirit levels.

Sensitivity for high precision levels is about several seconds of arc. Levels for surveying instruments generally have nominal sensitivities of 10, 15, 20, 60, 120 arc sec.

Electronic levels currently use a laser/liquid levelling system. The liquid is illuminated with a laser source, the reflected beam is focussed on to a four quadrant detector, thus the ratiometric output of the quadrant detector is directly proportional to the angle of inclination. The quadrant detector is described further in section 1.2.

Accuracy with this method is better than 0.2 seconds of arc, and generally used in first order geodetic levelling. Fig 1.3 below illustrates this technique:

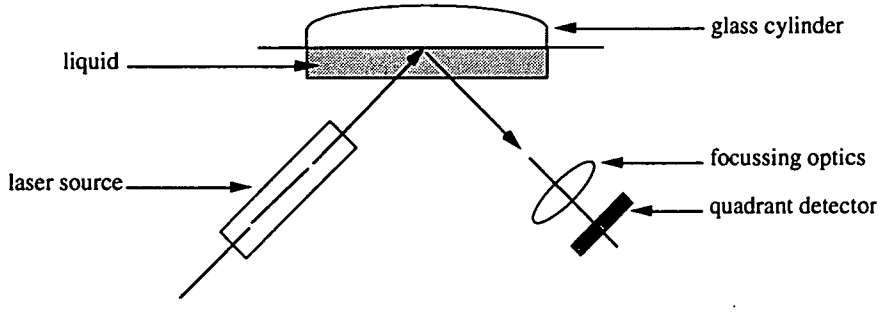


Fig.1.3 laser based levelling system

(b) Angular Measurements:

Modern electronic theodolites incorporate an electronic circle reading system. This consists of a photolithographically coded circle on a glass plate on which a series of opaque/transparent sections have been etched, with photoelectric detector and photodiode. Two square wave outputs are provided phased 90 degrees apart which determine the amount and direction of rotation, this requires a quadrature counter. Accuracy with this method is about 40 seconds of arc.

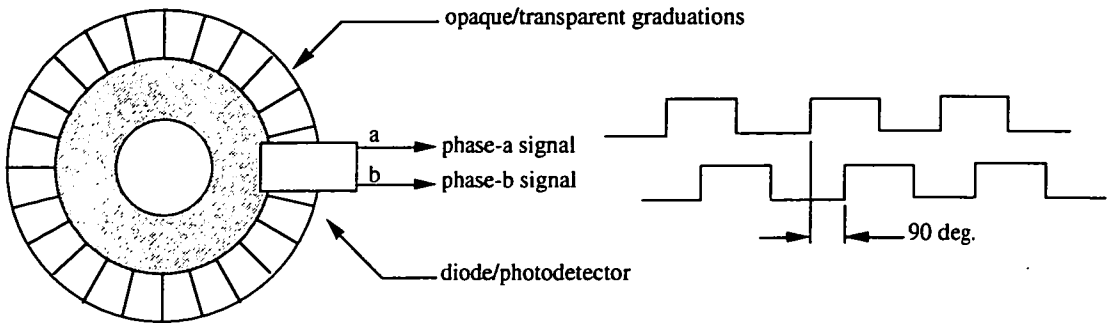


Fig.1.4 Electronic circle measuring system

More advanced theodolites use a moire' fringe pattern, this gives coarse and fine measurements, similar to Fig.1.4 above with sinuousoid rather than rectangular wave outputs.

The coarse component is obtained by squaring the sinuoid outputs from the encoder and feeding directly into a digital counter, this gives resolution to about 30 seconds of arc.

The fine measurement is obtained by interpolation ie: measuring the phase angle of the moire' signal to improve the resolution to 1% of the moire' period ie: about 0.3 seconds of arc.

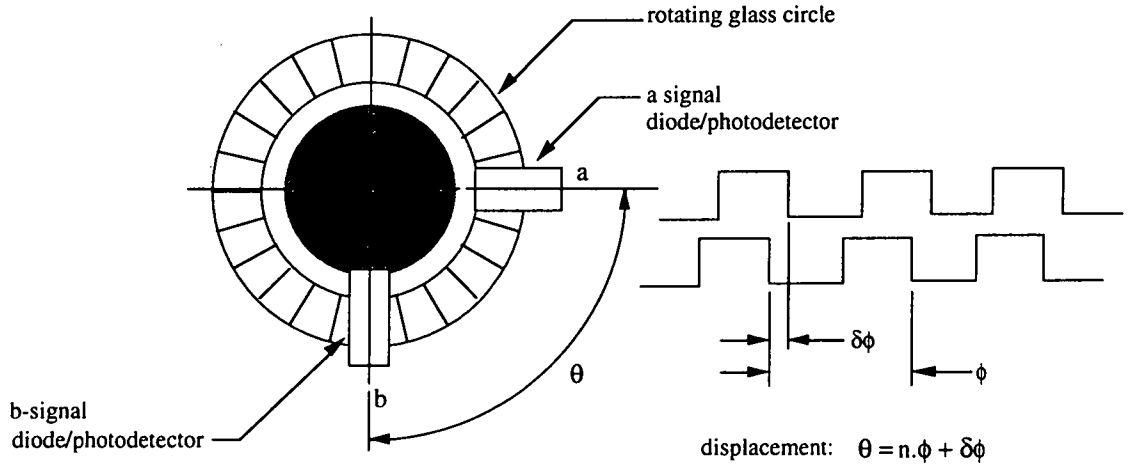


Fig.1.5 Electronic circle scanning measuring system

Yet another method currently used is based on a dynamic circle scanning technique. The circle is divided into N opaque/transparent segments. A drive motor rotates the glass circle past two photodiode/photodetectors, one set is fixed (A), the other (B) rotates on a separate plate where the theodolite is mounted. Refer to Fig.1.5.

The angle can be computed from measurements of fine and coarse data from the phase difference of the two signals. This is one of the most commonly used methods for the more accurate measuring theodolites.

The displacement (θ) is the integer count n multiplied by the line separation plus the phase difference between the two detectors providing the coarse measurement. Accuracy is about 0.5 arc seconds.
The method we have adopted for azimuth/elevation measurements is yet different from the others described above, (refer to section 2.5 on the method we used for fine positioning measurement).

(c) Distance Measurement:

Traditionally, distance measurement (DME) has been based on time-of-flight microwave radar techniques (tellurometers). Current methods are generally electro-optic based on either:

(a) phase difference techniques.
(b) time of flight (or pulse echo).

Microwave based distance measurement equipment (DME) requires an active transponder to provide a return signal. Advantages of microwave include increased range and operation over all weather conditions.

Laser based (LIDAR) DME equipment requires the use of a retro-reflector prism (corner cube reflector) or acrylic retro-reflector which return the incident beam along the same path. The main disadvantages of laser based DME systems is the dependence upon good atmospheric visibility and optical alignment with the reflector.

Phase Difference Method: This method uses an intensity modulated laser source. The return signal is phase compared to the transmitted signal, the distance is an integer number of wavelenghts n plus a fraction of wavelenght obtained from phase comparison:

$$2D = n_i + \Delta\lambda_i \qquad (1.1)$$

By taking a set of measurements, and switching wavelenghts, then both n_i and Δ may be estimated. Modern instruments selectively scan through different modulation frequencies and then solve for Δ and n_i by least squares. Range is generally up to 5Km, accuracy is of the order of +/-5mm.

The integration of EDM and electronic angle measurement is collectively known as Electronic Tacheometers.

Pulse Method: The return echo of a short high energy pulse is used in time-of-flight measurement, the distance is given by the equation:

$$2D = C.\Delta t \qquad (1.2)$$

where C=speed of light, dependent upon atmospheric conditions. This technique works well over long distances. Fig.1.6 below shows a typical corner cube reflector required by most electro-optic DME equipment:

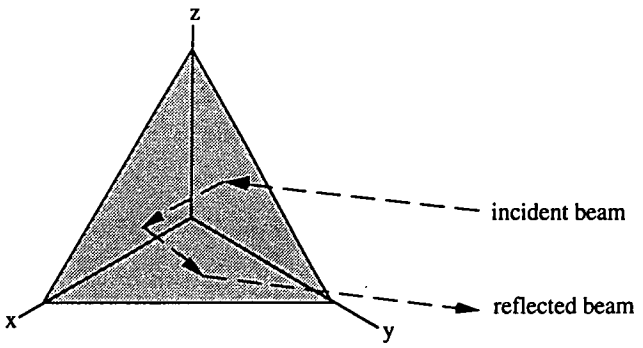


Fig.1.6 corner cube reflector

[1.2] Optical Tracking Systems:

Optical laser tracking systems measure the position of a retro-reflective target placed within the instrument's field of view with respect to its optical centre of field. This information is then fed back into a control system driving azimuth and elevation servo-motors, hence enabling the tracking system to re-centre itself on the target. One advantage of laser tracking systems is that they can be designed to be small, portable, compact and relatively lightweight.

Ample literature exists on this subject (see references 1.1, 1.3, 1.5 and 1.7 in the appendix 7.2A). One example is the star tracker on spacecraft systems, star trackers are passive instruments (rather than using active laser systems) in that they only detect the position of a point light source such as a bright stellar object. Tracking systems usually operate in two basic modes:

(a) target search mode.
(b) tracking mode.

In search mode, the instrument scans a prescribed space sector looking for a target, if the target is detected, then it is continuously tracked.

Normally angle tracking is required, although sometimes range tracking is also needed. Angle tracking implies continuous estimation of the target angle coordinates, this is typically achieved by the means of a mosaic sensor (eg: CCD or four-quadrant detector) which is described below.

(a) Target Position Estimation:

The optical tracker senses the target position with respect to its optical centre of field by focussing the target's image onto a sensor located on the focal plane. The two most common detectors are: the CCD array and the four quadrant detector.

Fig.1.7 below illustrates the basic principle. A laser beam is transmitted along the optical centre of the instrument, the return signal is focussed by a parabolic mirror onto a detector (either type), depending on the position of the retro-reflector will directly determine where on the detector the return laser signal is focussed.

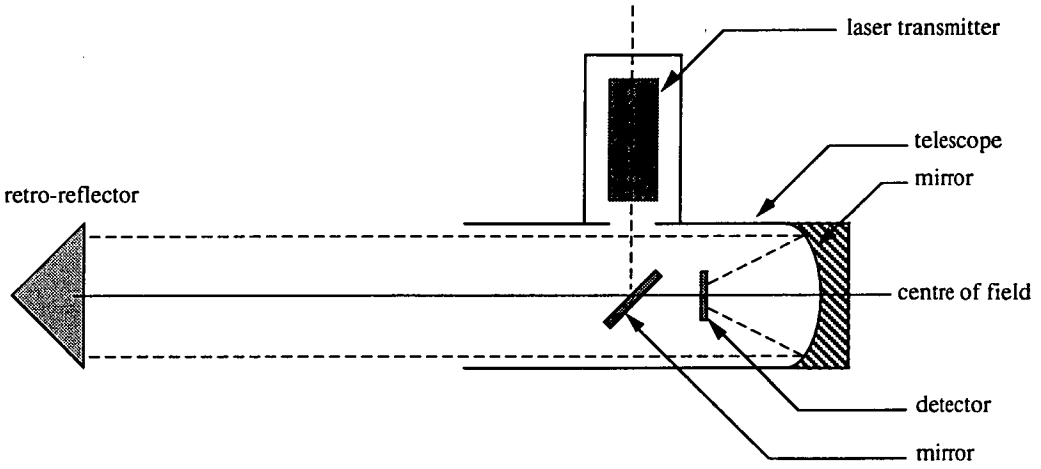


Fig.1.7 optical layout of typical tracking system

Four Quadrant Detector:

This being the simplest, the detector is of the type shown in Fig.1.8 below. It consists of four separate photo-sensitive detectors placed on a circular substrate. The ratiometric values between outputs of the four sensors determines the position of the retro reflector in azimuth and elevation planes. Accuracy with this system is about 20 seconds of arc. Equations 1.3A and 1.3B below give the relationship between position of the target as a function of the detector outputs.

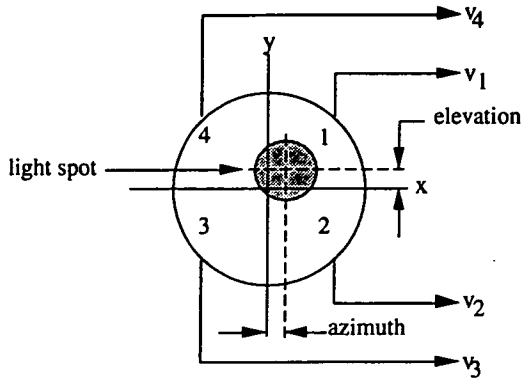


Fig.1.8 Four quadrant detector

Azimuth and elevation are given by: (where v_1, v_2, v_3, v_4 are the individual output voltages of each quadrant)

$$\text{AZIMUTH} \propto \frac{(v_1 + v_2) - (v_3 + v_4)}{\sum_1^4 v_i} \tag{1.3A}$$

$$\text{ELEVATION} \propto \frac{(v_1 + v_4) - (v_2 + v_3)}{\sum_1^4 v_i} \tag{1.3B}$$

CCD array:

Another common technique is to use a CCD array on the focal plane. The intensity distribution formed is usually a gaussian function over the two dimensional image plane, figure 1.9 below illustrates the technique. Equations 1.4A and 1.4B give the relationship between individual element output and azimuth/elevation.

$$x_p = \frac{\sum_{i=1}^{N} \sum_{j=1}^{N} v_{ij}\, x_i}{\sum_{i=1}^{N} \sum_{j=1}^{N} v_{ij}} \tag{1.4A}$$

$$y_p = \frac{\sum_{j=1}^{N} \sum_{i=1}^{N} v_{ij}\, y_j}{\sum_{i=1}^{N} \sum_{j=1}^{N} v_{ij}} \tag{1.4B}$$

Typical CCD detector array is shown in Fig.1.9, where x_p and y_p are the centroid of the light spot with respect to the lower left hand (datum) point (0,0), v_{ij} are the individual output voltages from each element of the array with the lower left hand corner (0,0) element voltage being v_{00} and N is the number of elements ie: NxN array.

This method is generally more accurate than the four quadrant photodetector system:

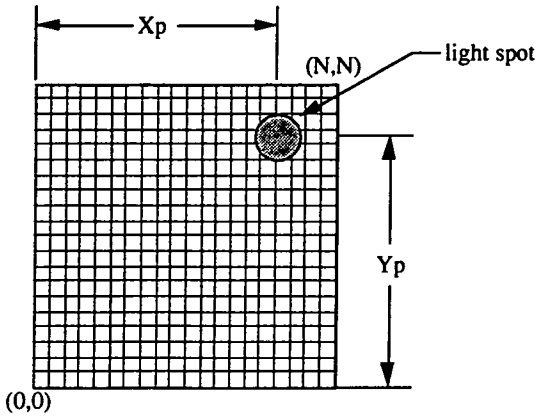


Fig.1.9 CCD array detector

Atmospheric disturbance such as scintillation caused by refractive index gradients, atmospheric scatter, vibrations in the optics, and measurement errors all contribute to apparent target jitter. For a moving target, velocity can be estimated in the presence of jitter by correlation or Kalman filter techniques.

Once the target position is estimated, the tracking system computer applies a correction signal to the servo motors. The correction signal rotates the instrument such that the centre of field of view is directly on to the target. Displacement is then measured by the main optical rotary encoders on the servo motors. This technique provides accuracy to only +/-1 step count on the rotary encoder.

An alternative method would be to rotate the servo motors by an exact integer number of steps and then measuring the position of the target with respect to the instrument's centre of field. The accuracy of this technique would then be dependent on the optical measurement system rather than the servo-motor encoders. The target's position is then given by:

$$x = n.\phi + x_p \tag{1.5A}$$

$$y = n.\phi + y_p \tag{1.5B}$$

where x and y are the overall displacement, n=number of integer steps, ϕ=step size, and (x_p,y_p) is the centre of the target given by Eqn.1.4A and 1.4B. This is the method which we have used in the Anglescan system.

[1.3] Anglescan: Principle of Operation:

The anglescan system is different from any of the conventional techniques described in the previous section. Fig.1.10 below illustrates the overall simplified system including the co-ordinates used:

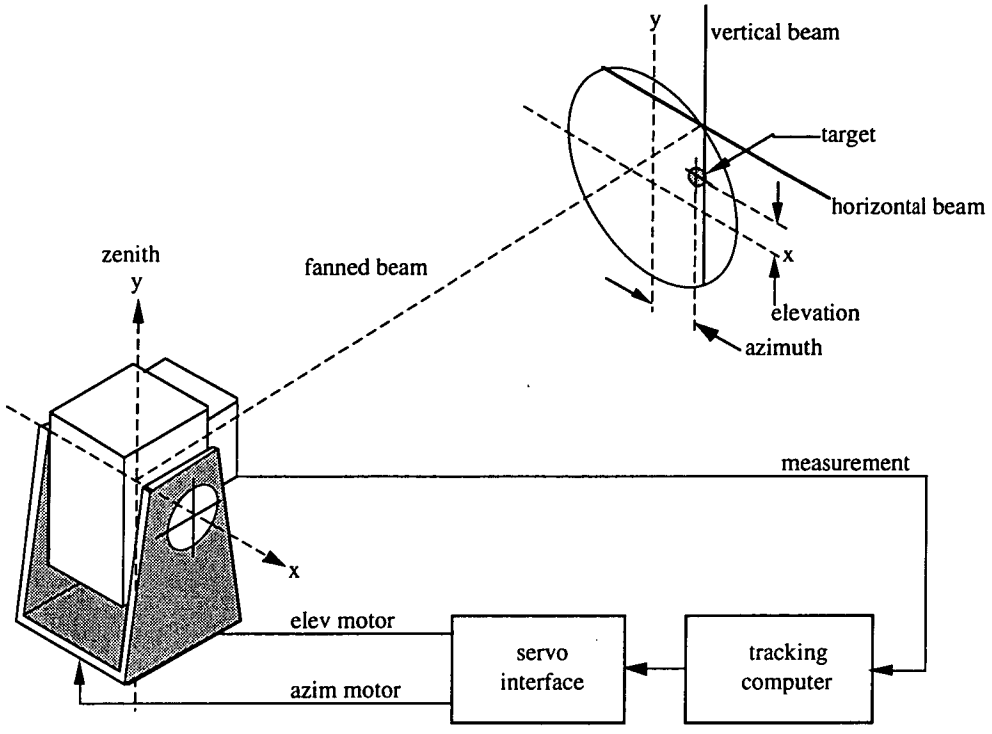


Fig.1.10 overall anglescan system

Referring to Fig.1.10, the instrument operates by projecting a cross, the two orthogonal lines forming the cross are labelled: vertical beam and horizontal beam. The instrument then rotates this cross in a circular path with a radius of 0.5 degrees, note that the vertical beam remains vertical always, and the horizontal beam remains horizontal always.

To detect and locate a target, it must be placed within the field of view of the instrument which is defined by the extent of the circle. Both the horizontal and vertical axes span at twice the diameter of the circle, that is 2 degrees. The cross rotates at 30 times per second.

Target position is measured each time the cross completes one full rotation, that is at a rate of 30 times per second. The target position data is subsequently used to generate a control error signal to the servo-motors such as to re-orient the instrument's centre of field directly on to the target. A detailed overview of the hardware is presented in chapter 2. The instrument has a maximum range of about 800 meters, the range is determined by the size of the reflecting target.

(a) Target Position Estimation:

The target position is measured by projecting a set of othogonal laser beams (diagram 1.11A, 1.11B, 1.11C, 1.11D below). The laser beams form a cross, the horizontal beam is labelled Hbeam, and the vertical beam is labelled: Vbeam, also refer to fig.1.10 above. The cross is then made to rotate along a circular path of radius equal to the field of view of the instrument. The laser source used is a 3mW HeNe visible (red) laser.

Referring to Fig.1.11A to 1.11D, two orthogonally projected beams forming a cross rotate along a circular path clockwise. Assume that an index pulse is generated by the rotary encoder (refer to fig.2.1) each time the centre of the cross passes through the top dead centre of the circle. The index pulse acts as a zero reference for the angular measurements of $\alpha_1,\alpha_2,\alpha_3,\alpha_4$.

Fig.1.11A: Initially the vertical beam (designated: Vbeam) intersects the target mirror from the left hand side, a return pulse is detected, the first measurement of α_1 is obtained from the optical rotary encoder.

Fig.1.11B: Next, the Horizontal beam (designated: Hbeam) intersects the target from above, producing an angular measurement of α_2 from the rotary optical encoder.

Fig.1.11C: Again, the vertical beam again intersects the target from the right hand side producing an angular measurement of α_3.

Fig.1.11D: Finally, the horizontal beam again intersects the target from below producing the last angular measurement of α_4.

The cross rotates 30 times per second, this value can be adjusted by varying the voltage on the motor. The sampling rate and position determination is set to 33 msec, also the loop time for closed loop control system analysis is 33 msec.

Measurements of α_1 and α_3 determine the target azimuth:

$$X_1 = R.\ \sin(\alpha_1) \qquad (1.6A)$$

$$X_3 = R.\ \sin(\alpha_3) \qquad (1.6B)$$

$$\text{AZIMUTH} = \frac{1}{2}(X_1 + X_3) \qquad (1.6C)$$

Measurements of α_2 and α_4 determine the target elevation:

$$Y_2 = R.\cos(\alpha_2) \tag{1.7A}$$

$$Y_4 = R.\cos(\alpha_4) \tag{1.7B}$$

$$\text{ELEVATION} = \frac{1}{2}(Y_2 + Y_4) \tag{1.7C}$$

equations 1.6C and 1.7C give geometric centre of target. Thus positioning accuracy is determined irrespective of the size of the target.

The method described above provides positioning accuracy to about +/-1 arc second. It has some inherent problems however, which are discussed in section 1.4.

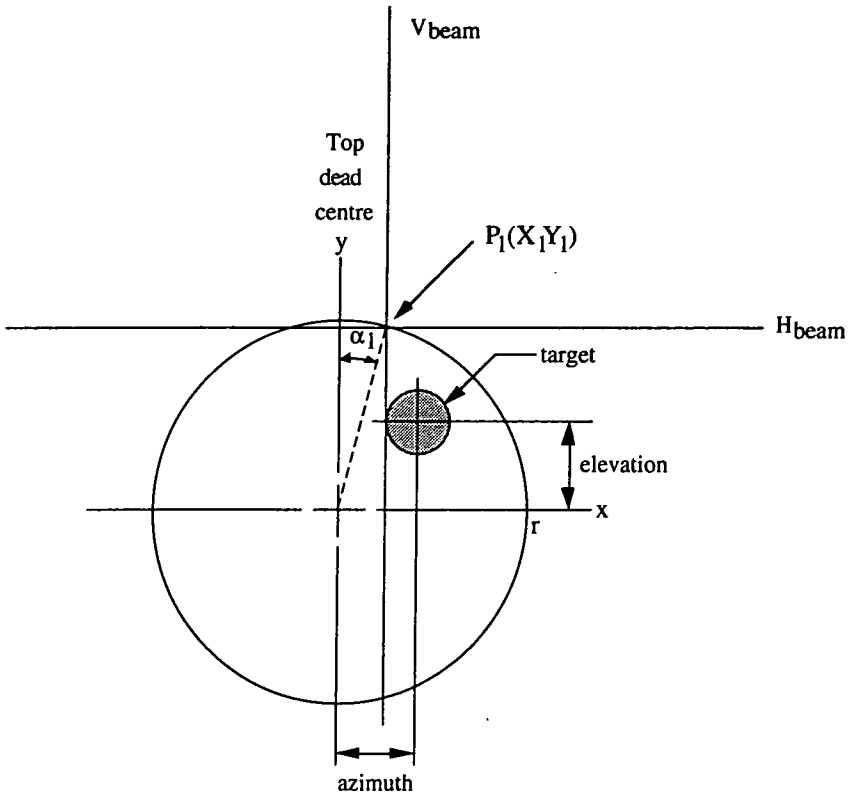


Fig.1.11A First intersection of target

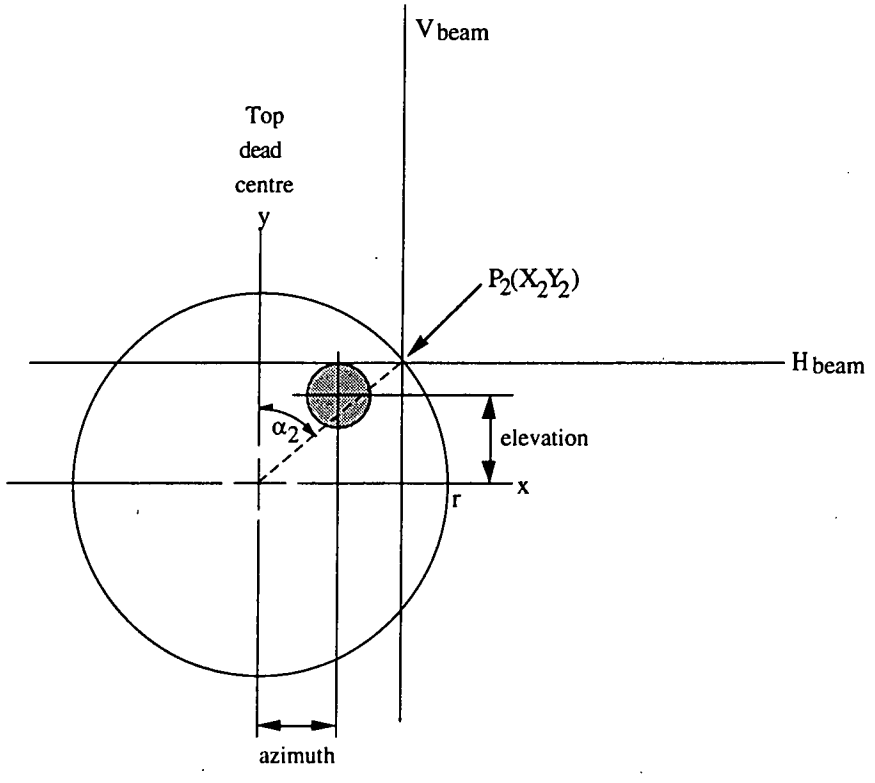


Fig.1.11B Second intersection of target

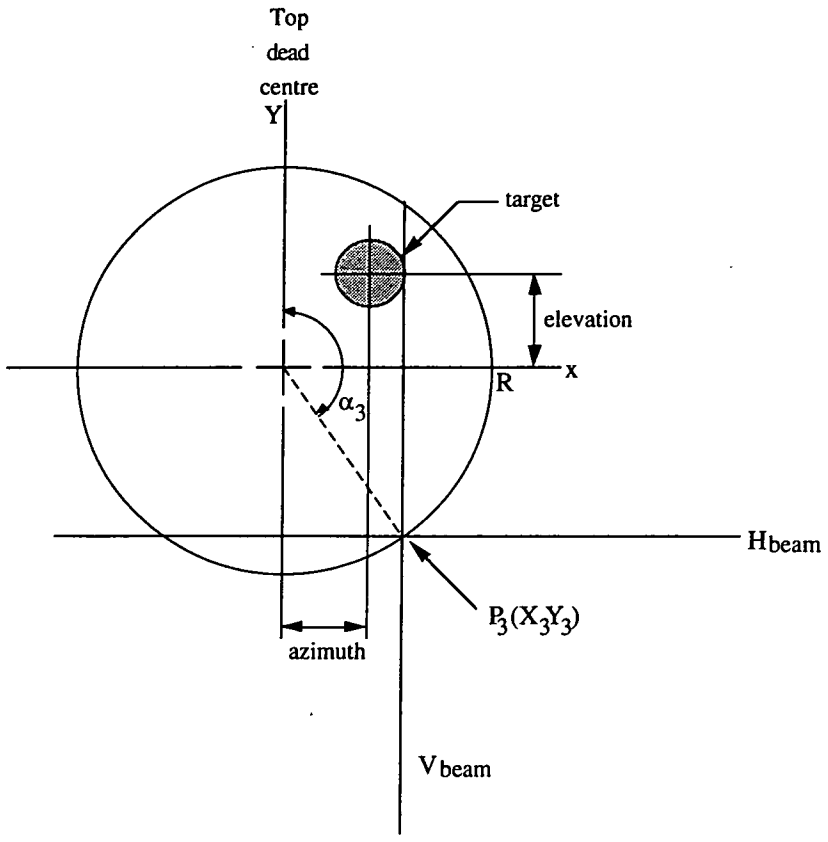


Fig.1.11C Third intersection of target

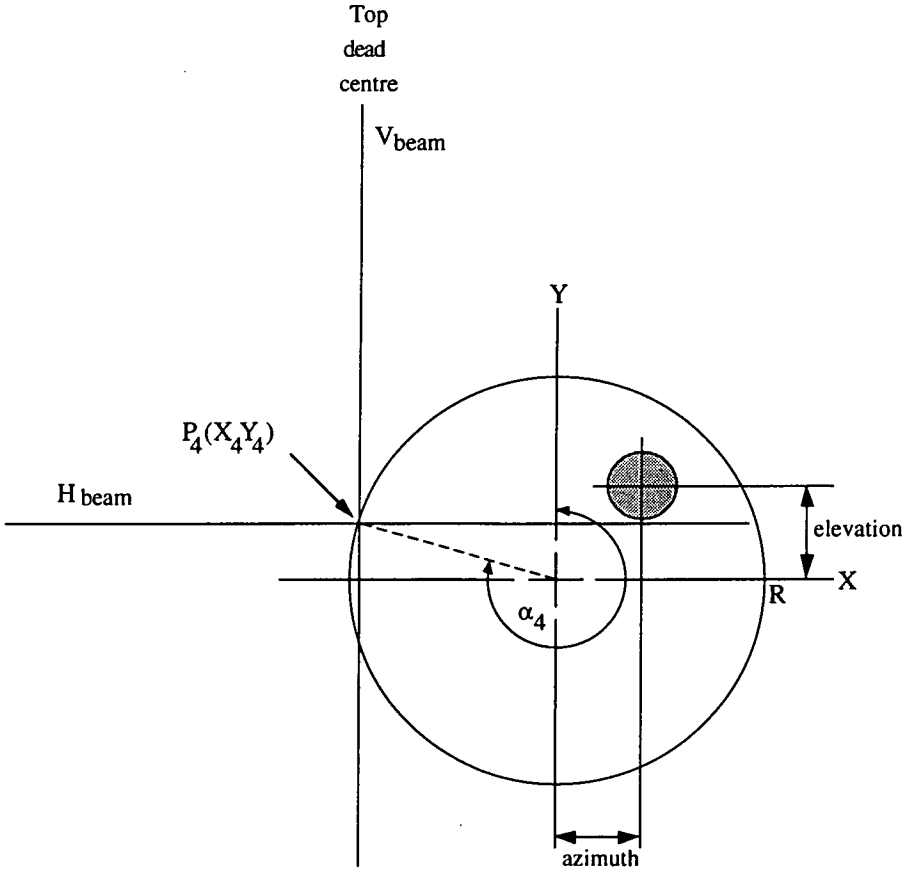


Fig.1.11D Fouth intersection of target

[1.4] Sources of Errors:

One of the advantages of this technique is the accuracy and simplicity in optical design, it has a number of pitfalls however including target positioning ambiguity, beam orthogonality, location of index pulse, and relative motion of target with respect to the tracker. Most of these problems can be resolved with calibration (refer to chapter 5) or from simple geometric tests in the anglescan software.

(a) Target Positioning Ambiguity:

Target positioning ambiguity can result when the target is placed in either quadrant 3 or 4 of the circle, Refer to Fig.1.12 below:

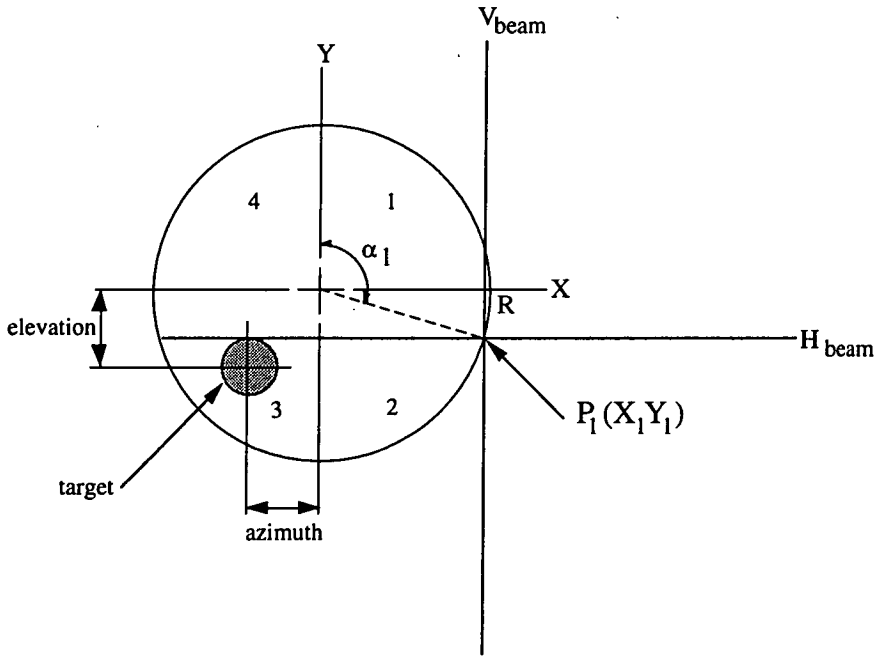


Fig.1.12 target ambiguity condition

When the target is located in either one of quadrant 3 or 4, then the first time the beam intersects the target is when the horizontal beam passes through from above giving the first measurement of α_1. The data sequence is then HVHV as opposed to the previous description (see Fig. 1.11A) where the vertical beam strikes first producing the data sequence of VHVH.

When the condition demonstrated by Fig.1.12 occurs, the computation of azimuth and elevation requires the measurements to be reversed in the equations thus:

$$x = \frac{R}{2}[\sin(\alpha_2) + \sin(\alpha_4)] \quad (1.8A)$$

$$y = \frac{R}{2}[\cos(\alpha_1) + \cos(\alpha_3)] \quad (1.8B)$$

Despite this problem, position ambiguity can be easily resolved by a simple geometric test conducted by the software.

(b) Beam Orthogonality:

The projected horizontal and vertical beams cannot be guaranteed to be truly othogonal. Additionally the horizontal beam cannot be assumed to be truly horizontal (ie: parallel to the horizon), and the vertical beam cannot be assumed to be truly vertical.

Fig.1.13 below illustrates the problem, the errors are caused by optical and mechanical alignment. Generally the values of ϕ_v and ϕ_h are to within +/-5 degrees. To maintain constistent instrument accuracy, we shall determine the horizontal and vertical beam error during the calibration stage (discussed further in chapter-5).

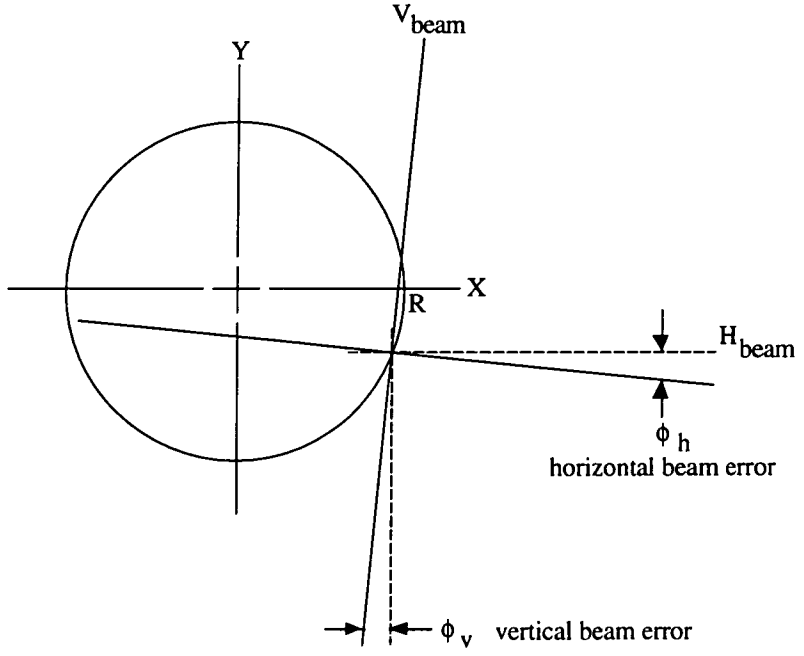


Fig.1.13 Horizontal and vertical beam errors

For applications requiring alignment or levelling, this error condition should be resolved accurately.

(c) Index Pulse:

So far, the assumption has been that the reference (zero index pulse) occurs at the exact top dead centre of the circle. This is however not the case as the index pulse can occur anywhere within the 360 degrees depending on the mechanical assembly of the optical rotary encoder. Although the position of the index pulse remains constant (mechanically fixed) it is an unkown parameter which needs to be determined from instrument calibration. As the index pulse is non zero, equations (1.8A) and (1.8B) are subsequently modified to accommodate for this condition. Referring to quations 1.9A and B below, we define Ip as the angular position of the index pulse with respect to the top dead centre:

$$x = \frac{R}{2}[\sin(\alpha_2 - I_p) + \sin(\alpha_4 - I_p)] \quad (1.9A)$$

$$y = \frac{R}{2}[\cos(\alpha_1 - I_p) + \cos(\alpha_3 - I_p)] \quad (1.9B)$$

Fig. 1.14 illustrates the condition where the index pulse occours in the fouth quadrant.

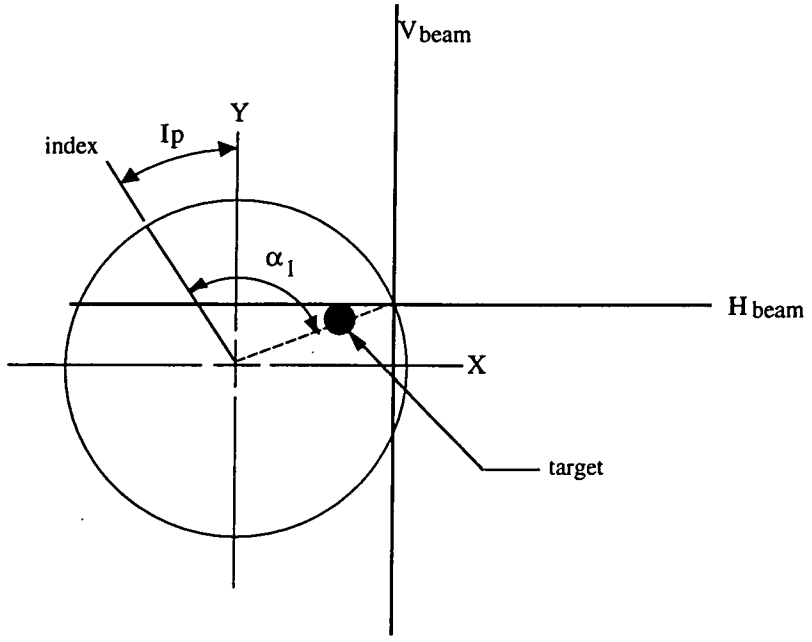


Fig.1.14 index pulse offset

(d) Relative Motion of Target:

The relative motion of the target with respect to the instrument introduces positioning calculation errors at the end of the sampling instance. The end of the sampling occurs on the next index pulse.

Refering to Fig.1.15 below, and considering only motion effects on the azimuth position calculation: when the target is at **a** then the vertical beam intersects at P1.
By the time the beam intersects the target at P3, the target would have moved to point **b**. Finally, when the beam reaches the top dead centre to complete the sampling cycle, all four measurement are available, the target may have moved to point **c**.

As the calculation of azimuth is a geometric average between measurements obtained at P1 and P3, the tracking computer will thus estimate the target azimuth to be the midway between P1 and P3 ie: target AZIMUTH=$(x_1+x_3)/2$. This error is by far the most important while tracking a moving target and can introduce instability in the closed loop control system.

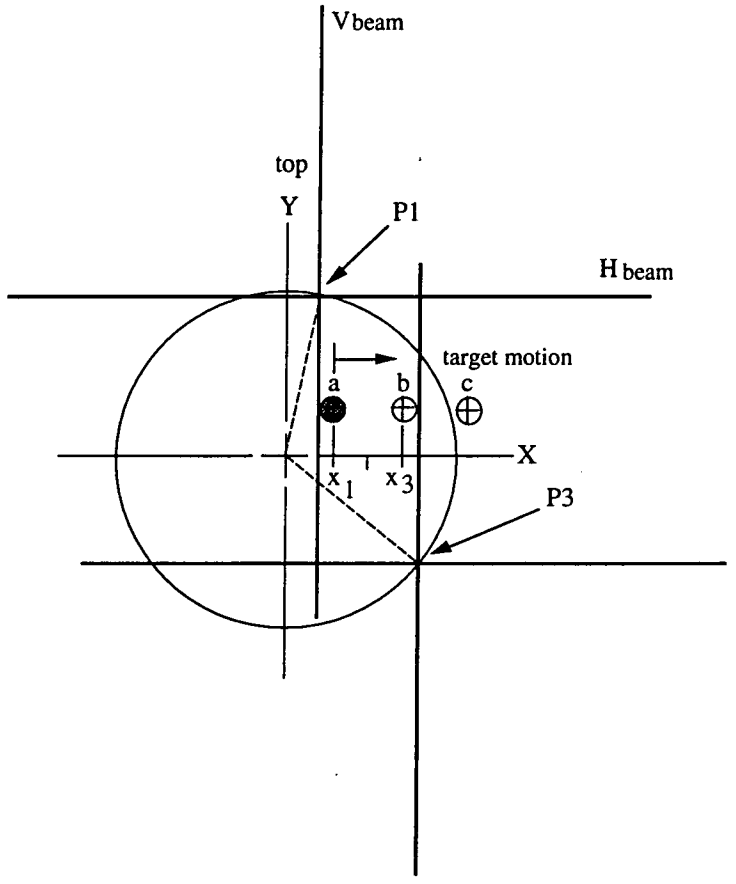


Fig.1.15 apparent target motion within fov

In section 4.3 we introduce the effects of apparent target motion into the control system model and show that the system's overall margin of stability is actually reduced due to this measurement process. Intuitively however, we can see that the control loop can overcompensate when the target is actually closer to the centre than the calculation indicates, or undercompensate when the target is actually further from the centre than the calculations indicate.

[1.5] Chapter Summary:

In this chapter we presented some of the basic concepts behind the construction of an electronic theodolite including the three main uses of a theodolite: (1) Levelling, (2) Angular measurement, and (3) Distance measurement. We have shown the method by which the theodolite is able to achieve each of these objectives.

We also discussed the basic construction of an optical tracking system, and the means of determining the angular position of a target placed within the field of view of the instrument.

We have finally introduced the anglescan system and the principle behind the measurement technique. The anglescan system essentially comprises of a combination of the two instruments described above but with a totally different technique for the determination of the target position.
Such instruments currently do exist as commercially available theodolites with target tracking capabilities, however the anglescan system is the only one with this kind of measurement technique and subsequently most of the work on the instrument would be new and innovative.

In the following chapters we will discuss the different aspects of the anglescan system and the hardware/software involved for such instrument.

Chapter-2

Hardware and Software Design:

[2.0] General Overview:

Fig.2.0 below shows a detailed assembly of the main components of the anglescan tracking system. The system consists of two parts: (a) the Anglescan optical and mechanical system, generally mounted on a tripod and (b) the Anglescan electronic and software system. The electronic system consists of 6 boards interconnected via a dedicated backplane bus, and power supply. There is also a front panel LCD display and keyboard.

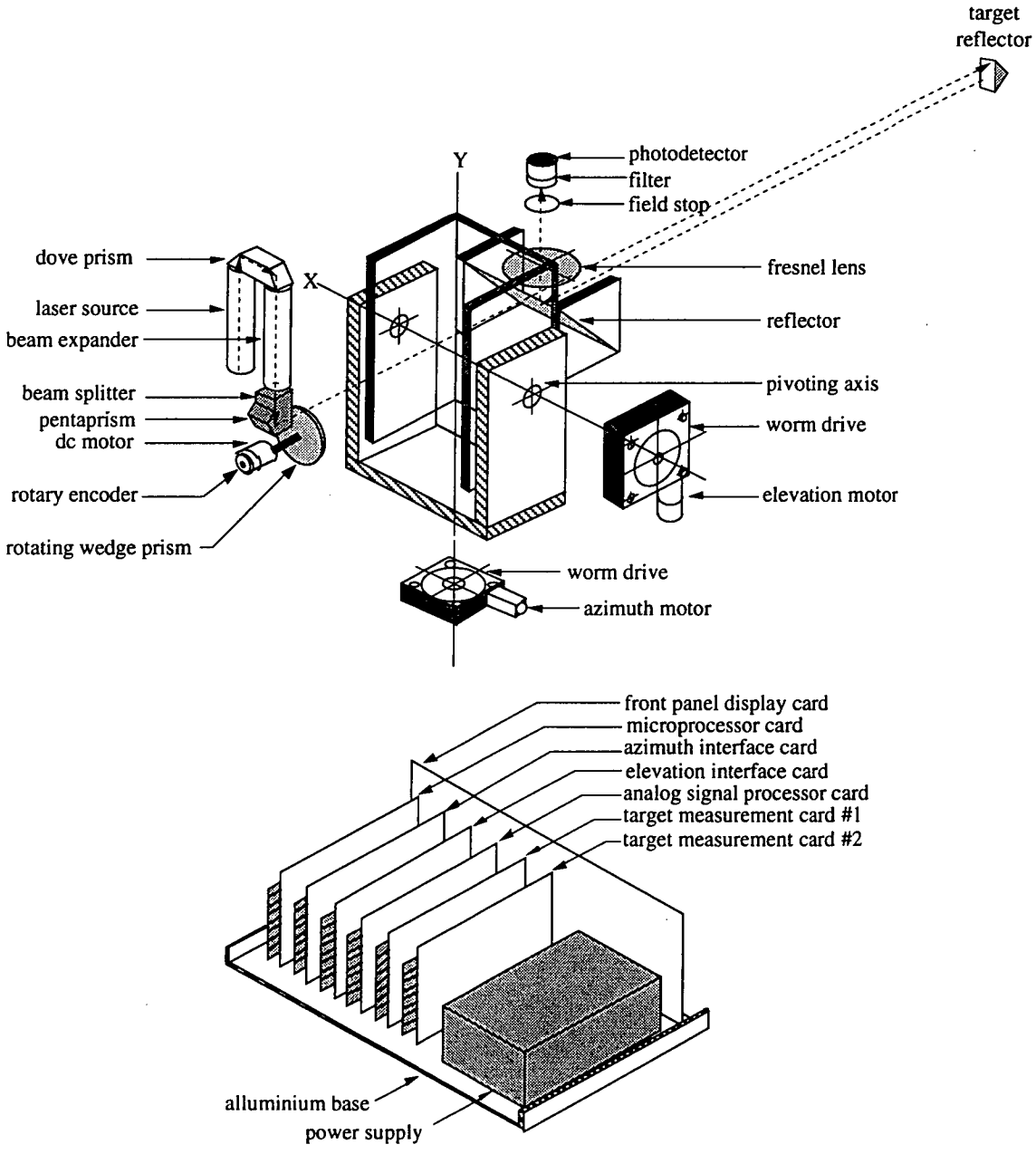


FIG.2.0 Anglescan System Overview

(a) Optical System

Referring to fig.2.0, a 3 mW HeNe laser source is used, the output of the laser is reflected by a dove prism into a beam expander. The laser beam is thus expanded from a single point into a line. The output of the beam expander is fed into a beam splitter a pentaprism and a rotating wedge prism. This has the overall effect of projecting a rotating cross (refer to fig.2.1).
When the laser cross strikes a target retro-reflector, a return pulse is produced. The return pulse is reflected into a fresnel focussing lens, a field stop and interference filter, and photodetector placed on the focal plane. The output from the photodetector is fed into the analog signal processing card.
Servomotors are mounted on the azimuth and elevation axis. Each servomotor consists of a 24V DC motor and worm drive assembly, it also includes an optical rotary encoder for positioning measurements.

(b) Electronics System:

There is a separate power supply for the HeNe laser which generates about 7KV nominal output voltage mounted near the laser. The laser output power is typically 3mW. The existing laser will eventually be replaced by a semiconductor infrared laser which operates at much lower voltages (3Vdc). Figure 2.2b illustrates the second part of the anglescan system which is the electronic system and interface to the optical and servo-system. The electronic system consists of the following cards:

(1) Front panel keypad and LCD display card.
(2) Z80 microprocessor card with 32K RAM, 32K EPROM, and dual UART.
(3) Azimuth servomotor interface card with azimuth encoder positioning measurement.
(4) Elevation servomotor interface card with elevation encoder positioning measurement.
(5) Analog signal processor card for detecting and processing return pulses.
(6) Return pulse measurement card #1.
(7) Return pulse measurement card #2.

(c) Power Supply:

The power supply consists of two separate mains step down transformers with one supplying +12V to the high voltage inverter used by the HeNe laser. The second transformer supplies +/-18V unregulated to all the individual cards via the backplane bus, each board has its separate voltage regulators typically +5V and +/-12V for CMOS and analog supplies.

[2.1] Optical Transmission System:

The optical transmission system is shown in Fig.2.1. It consists of a 3mW HeNe visible red laser and a dove prism reflecting the beam into a beam expander. The purpose of the dove prism is to allow the laser and beam expander to be mounted side by side thus reducing the overall physical size of the instrument.

The point beam passes through a collimator and beam expander thus producing a fanned beam with a divergence of 2.0 degrees along one plane, the output of which is then separated into two equal intensity beams via a beam splitter. The first half forms the vertical component of the projected cross, whereas the second half is rotated by 90 degrees with a pentaprism and is then projected to form the horizontal component of the cross.

The rotating wedge prism is a circular disk made of optical glass attached concentrically to a dc motor. When the wedge prism is rotating, the projected cross rotates along a circular path, the radius of which is a function of the refractive index and geometry of the wedge prism. Currently the prism causes a deflection angle R of 0.5 degrees radius.

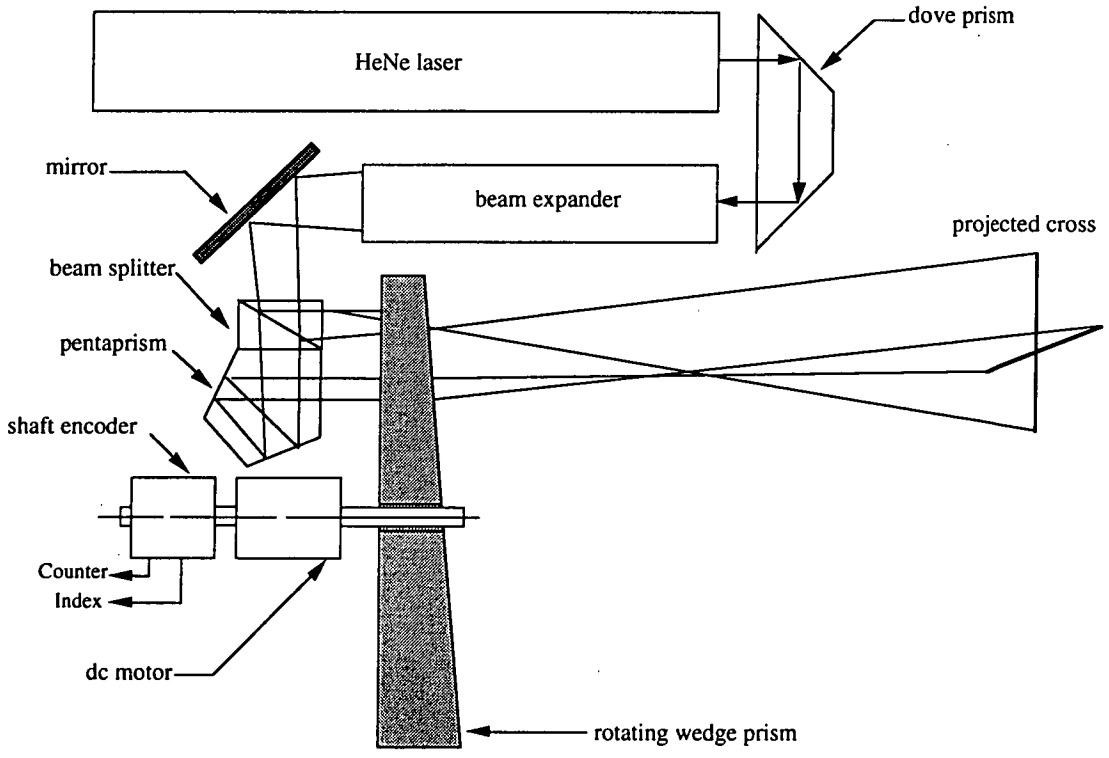


Fig.2.1 Anglescan optical system

The motor and gearbox combination rotates the wedge prism at a fixed rate of 30 Hz. The velocity can be controlled by adjusting the DC voltage on the motor. A shaft encoder is mounted on the motor.

The shaft encoder measures the position of the cross along the circle with a resolution of 10,000 counts per revolution, it also generates a reference index pulse once per revolution. This index pulse is used as a sampling reference to reset and then restart the four counters which measure the values of $\alpha_1,\alpha_2,\alpha_3,\alpha_4$. Refer to section 2.4. and fig.1.11A,B,C,D.

[2.2] Photo-Detector Optics:

The detector optics consists of a large mirror (reflector), a Fresnell focussing lens, a field stop and a narrowband interference filter with the passband centre at the laser wavelenght. This effectively removes a high proportion of ambient background light. The raw signal is detected by a hybrid photodetector-amplifier which provides low noise and low output impedance.

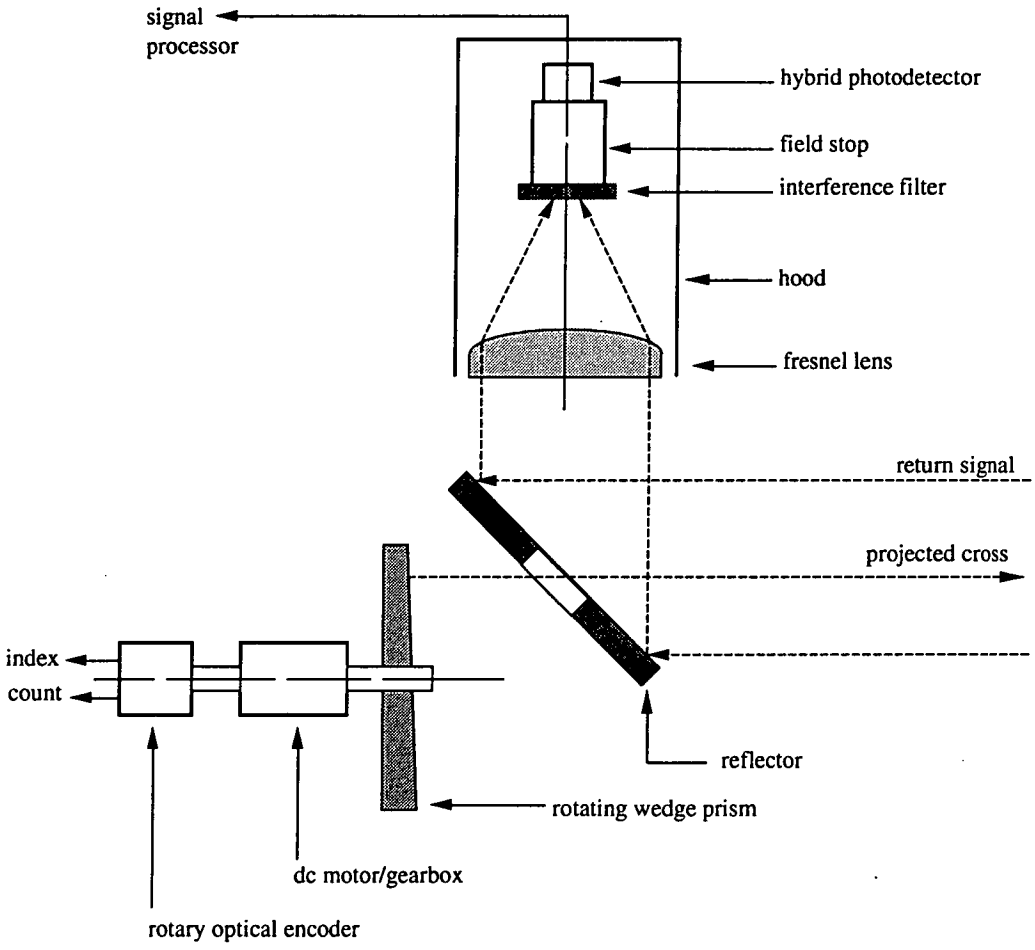


Fig.2.2 Photodetector optics

This signal is then fed to an analog signal processor board (described in section 2.6). Fig. 2.2 illustrates the design of the detector optics.

The photodetector signal is nominally 5 mV peak to peak amplitude at a maximum range of about 800 metres. The noise level has approximately the same peak amplitude. This is the limiting factor in determining the maximum range and signal detection at this range is unreliable. At a range of 8 meters, the signal amplitude is about 500mV peak to peak. The signal level attenuates approximately in inverse proportion to distance.

[2.3] Block Diagram of Electronic System:

The hardware is configured around a dedicated backplane bus and microprocessor to which custom boards have been interfaced. Figures.2.2b and 2.3 show the overall simplified schematic diagram of the electronic system.

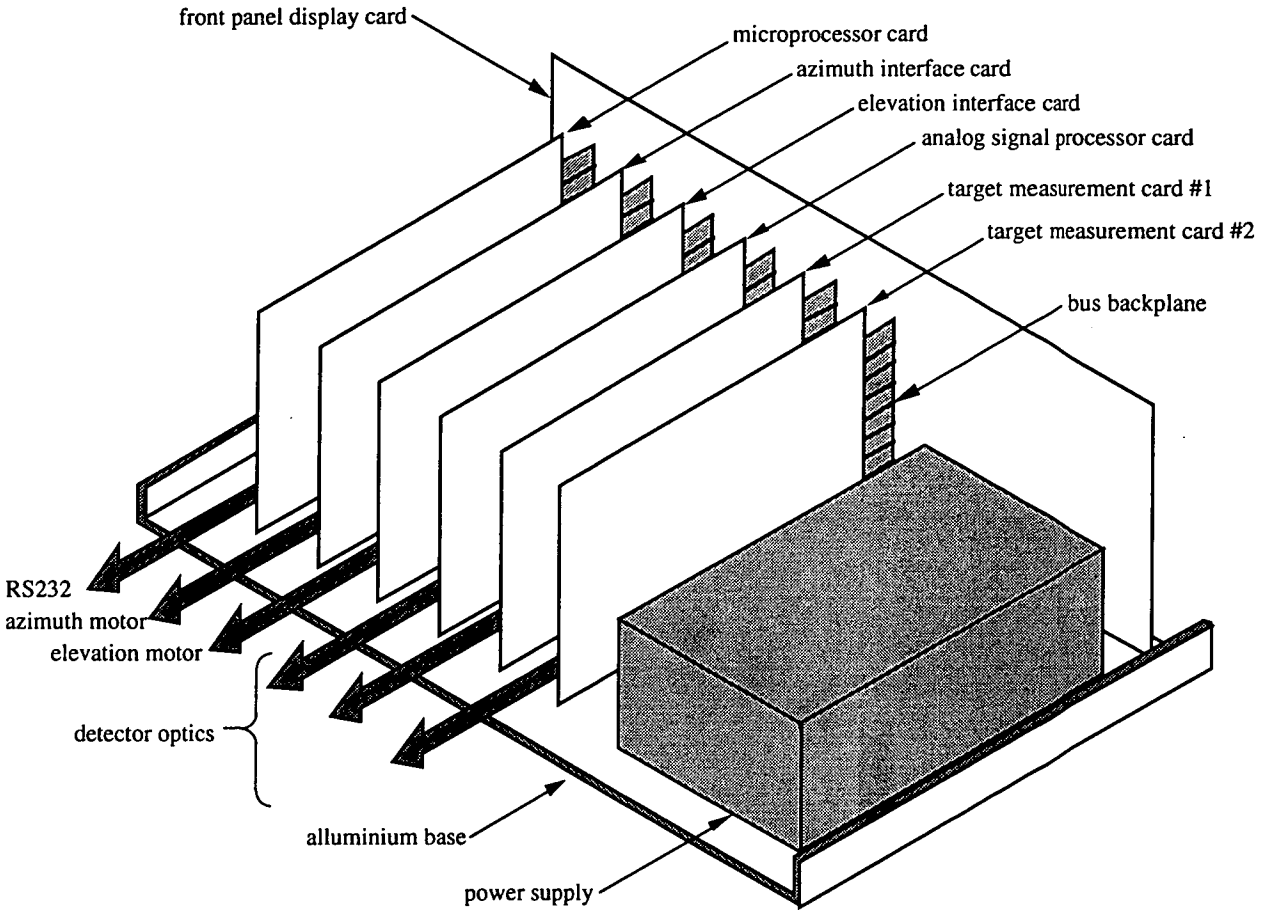


Fig.2.2b Anglescan Electronic System - Mechanical Assembly

The electronic system consists of six identical size boards connected via a bus backplane which uses ribbon cable and 40 pin-IDC connectors, a front paned keypad and display card. The dual power supply comprises of +12V and +/-18V (2A) unregulated outputs.

The order of the placement of the boards is not important. The microprocessor board is placed leftmost to allow easy removal and insertion of EPROMS during software testing and development. The front panel card is not used because it is simpler to use the RS232 available on the microprocessor card to communicate directly to a serial terminal. The purpose of the front panel card is to allow the instrument to be fully portable for field testing.

Schematic circuit diagrams and PCB artwork were all done with PROTEL AUTOTRAX PCB and PROTEL SCHMATIC schematic editor. Detailed copies of circuit diagrams, bus wiring diagrams and component overlays is given in the appendix 7.5. The function and purpose of each of the above boards is briefly outlined below.

(a) Z80 Microprocessor board:

The microprocessor card contains 32K RAM and 32K EPROM memory. It implements the closed loop control system tracking algorithms, target search and acquisition routines, calibration routines, debugging/monitor routines and host terminal interface. The software is developed in HITECH-C. The source code is about 100 pages of high level C and utilizes the full 32K EPROM memory. The software listing is not included in the thesis because of its size, but a brief description is given in sections 2.9 and 2.10. The user menu interface is described in appendix 7.7A. The microprocessor card has a dual UART for serial communications to a host terminal.

(b) Analog Signal Processor:

The analog signal conditioning board is used for processing laser return signals. It includes a software controlled AGC stage (range tracking), signal strength detection, second order active filters, software programmable threshold detector, and ambient background light level detector, and digital de-glitching filters. The input return laser pulses are received from the photodetector and subsequently fed into the analog signal processor board. The output is a TTL compatible signal.

(c) Target Measurement Card #1:

The target measurement card measures the four parameters: α_1, α_2, α_3, α_4, refer to figure 1.11A,B,C,D. This board measures from the index pulse to the leading edge of each return pulse. The laser return pulses are fed from the output of the signal processor card into the position measurement card. Refer to fig.2.4A.

(d) Target Measurement Card #2:

The second board measures from the index pulse to the trailing edge of each return pulse, this way, the precise centre of the return pulses can be calculated in software. It provides the four measurements: $\{\beta_1, \beta_2, \beta_3, \beta_4\}$. Refer to figure 2.4b.

(e) Elevation Servo Interface Card:

Open loop servo motor driver board using Pulse Width Modulation. It includes a 16-bit quadrature counter for coarse elevation positioning measurements obtained from the rotary positioning encoder on the servomotor output shaft. A detailed description is given in section 2.5. The board also includes fine positioning measurement logic to determine elevation angles with greater precision than the available measurements from the optical rotary encoder. The board is fully programmable under software control.

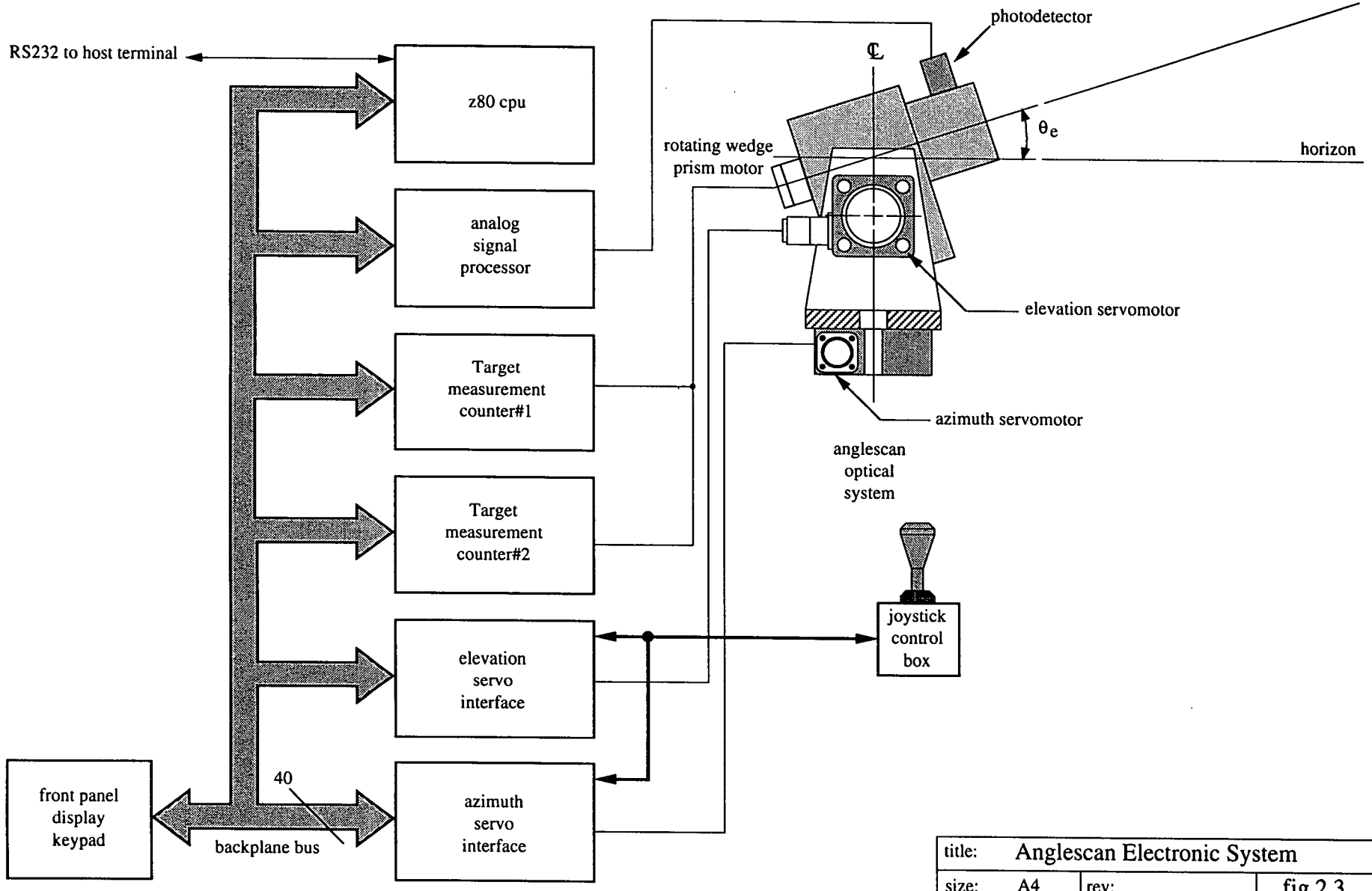
(f) Azimuth Servo Interface Card:

Open loop servo motor driver for the azimuth servo. Identical to the elevation interface board with different I/O port addresses. Also refer to section 2.5. The purpose for developing the azimuth and elevation boards separately is because of the complexity of each board.

(g) Front Panel Card:

This card contains the front panel keypad, LCD display, and PCB mounted front panel DB25 connectors for serial commnunications to the microprocessor card.

Additionally, the system incorporates a joystick control to enable the instrument to be manually rotated in both azimuth and elevation. Fig.2.3 on the following page illustrates the general interconnection and wiring of the anglescan optical system and the various hardware components of the anglescan electronic system.



title:	Anglescan Electronic System		
size:	A4	rev:	fig.2.3
file:			sheet of:
date:			drn by:

[2.4] Target Measurement Card:

The Target measurement cards (refer to Fig.2.3) denoted as: TARGET MEASUREMENT COUNTER#1 and TARGET MEASUREMENT COUNTER#2 measure the location of the target within the field of view of the instrument by providing the four pairs of angular measurements: $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$ and $\{\beta_1, \beta_2, \beta_3, \beta_4\}$. Refering to Fig.2.4a and b, the output from the return laser signal processor card generates four rectangular pulses per rotation, each rotation is referenced by an index pulse. Originally, a software counter was used, but it was found to require excessive amounts of CPU time, it was consequently fully hardware implemented in discrete logic to operate independently of the microprocessor.

Two separate counter boards are required: **COUNTER#1** and **COUNTER#2**. The first counter board measures from the start of the index pulse to the leading edge of each of the four return pulses as shown in Fig.2.4a. The second counter board measures from the start of the index pulse to the trailing edge of each of the four return pulses respectively. Each counter board contains four 16 bit binary counters (see fig.2.5).

Description:

The output from the rotary encoder on the rotating wedge prism generates 10,000 pulses per revolution and an index pulse. The rising edge of the index pulse latches the previous counter measurements into octal tristate latches. The falling edge of the index pulse resets and restarts all the four counters on each board simultaneoulsy.

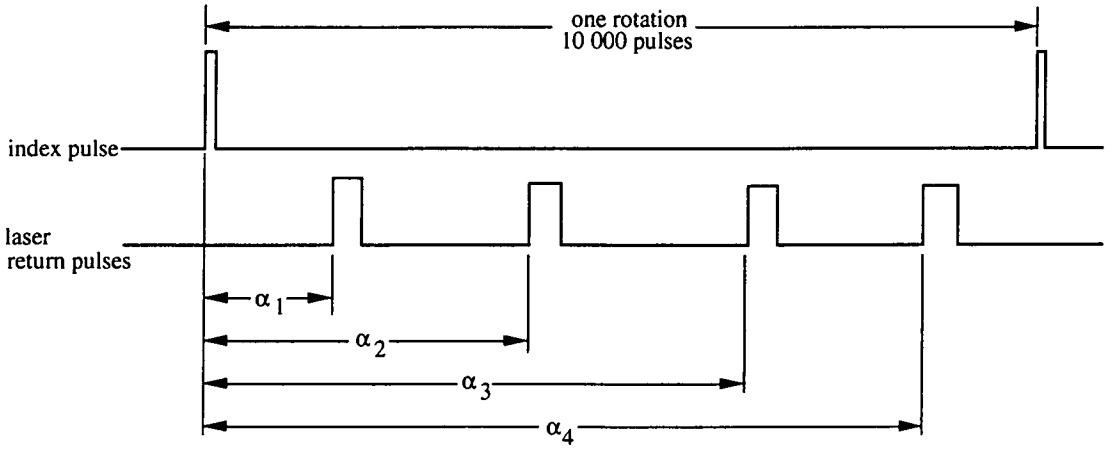


Fig.2.4a Position Measurement Counter #1 Board

The use of two counter boards enables consistent positioning accuracy to be obtained regardless of the distance of the target (and hence the width of the return pulse).

Similarily, the target measurement **COUNTER#2** board measures the number of pulses between the occourence of the index pulse and the trailing edges of each of the return pulses as show in Fig.2.4b below:

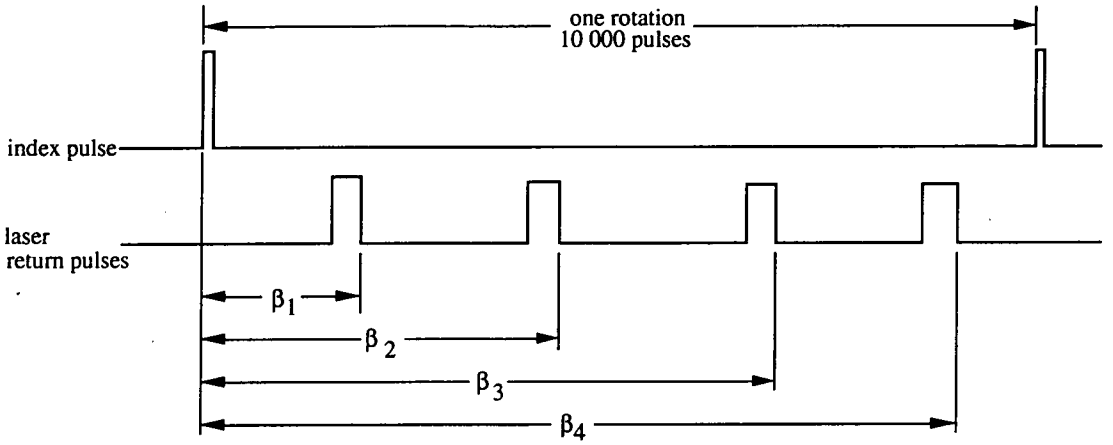


Fig.2.4b Position Measurement Counter #2 Board

The two boards **COUNTER#1** and **COUNTER#2** are identical in design, except for the edge triggering. The principle of operation of the measurement boards is illustrated by figure.2.5.

Refering to figure.2.5, the rising edge of the index pulse serves to latch the four 16-bit binary counters. The falling edge of the index pulse clears all four 16-bit binary counters, the counters are incremented from the rotary encoder output which is a quadrature signal (Phase-A and Phase-B), each time a return pulse is detected, counters are progressively disabled (gated off) from counting.

The leading edge of the index pulse is used to store the contents of the four binary counters into four 16-bit tri-state latches, data is available to the computer at any time excluding the period in which data is being transfered from the counters to the output latches.

If more than four return pulses have been detected (eg: due to stray light from some reflective object in the field of view), the CPU will detect an OVERCOUNT condition and discard all four measurements. Conversly, if less than four measurements are available (due to a weak return signal or the beam has been inadvertently interrupted) then the CPU will detect an UNDERCOUNT condition and discard all the four measurements. During target tracking, such conditions can be ignored and the computer may use previous data or a predictive algorithm.

When no measurements are available ie: ZERO-COUNT condition, the target is assumed to be out of range (too far) or outside the instrument's field of view.

Figure 2.5 illustrates the basic components of the target measurement cards, figure 2.5a shows a more detailed block diagram.

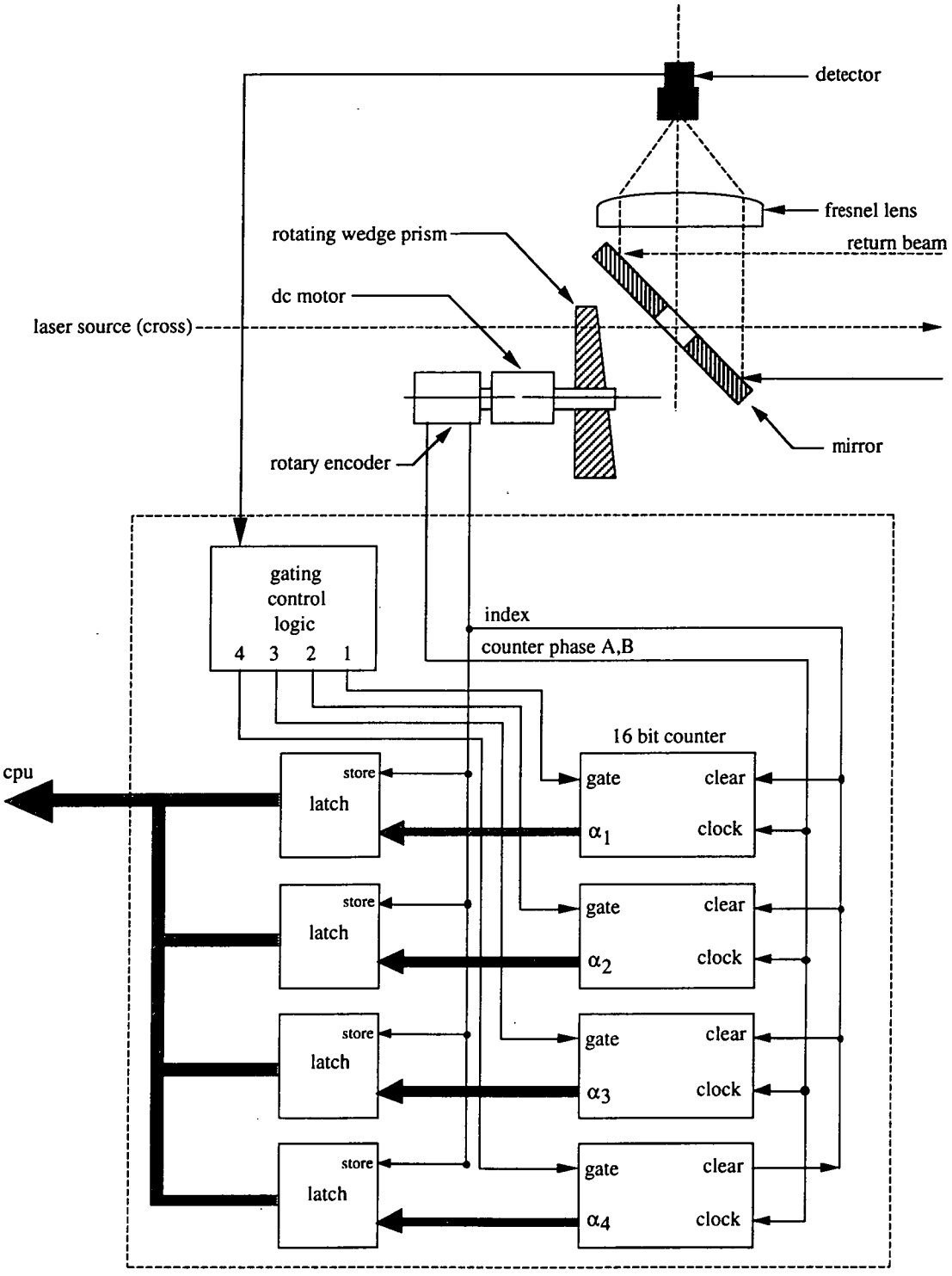


Fig.2.5 Target measurement board.

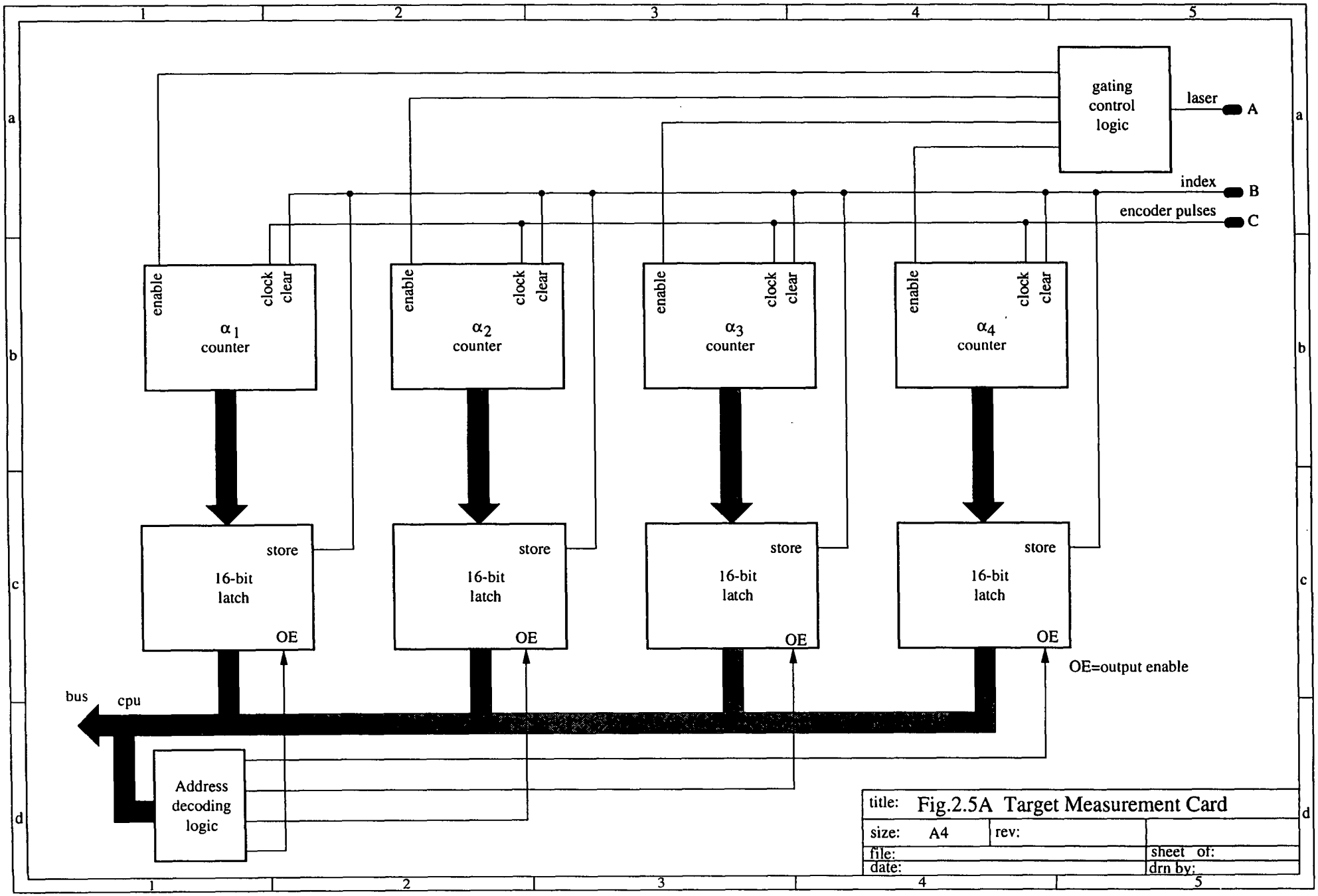
If the retroreflective target is moving further away, then the width of each return pulse becomes narrower. The width is approximately proportional to the ratio or R_m/d, where R_m=mirror radius and d=range.

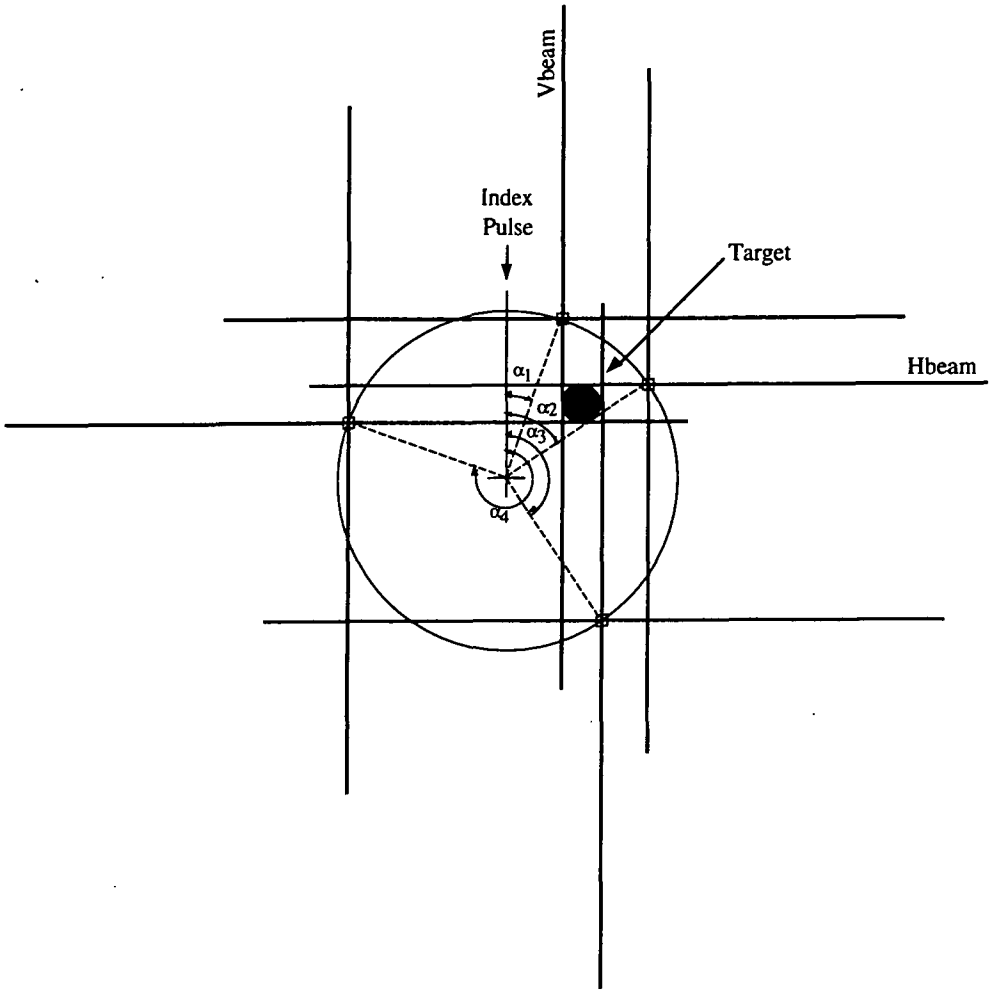
Circuit Diagram:

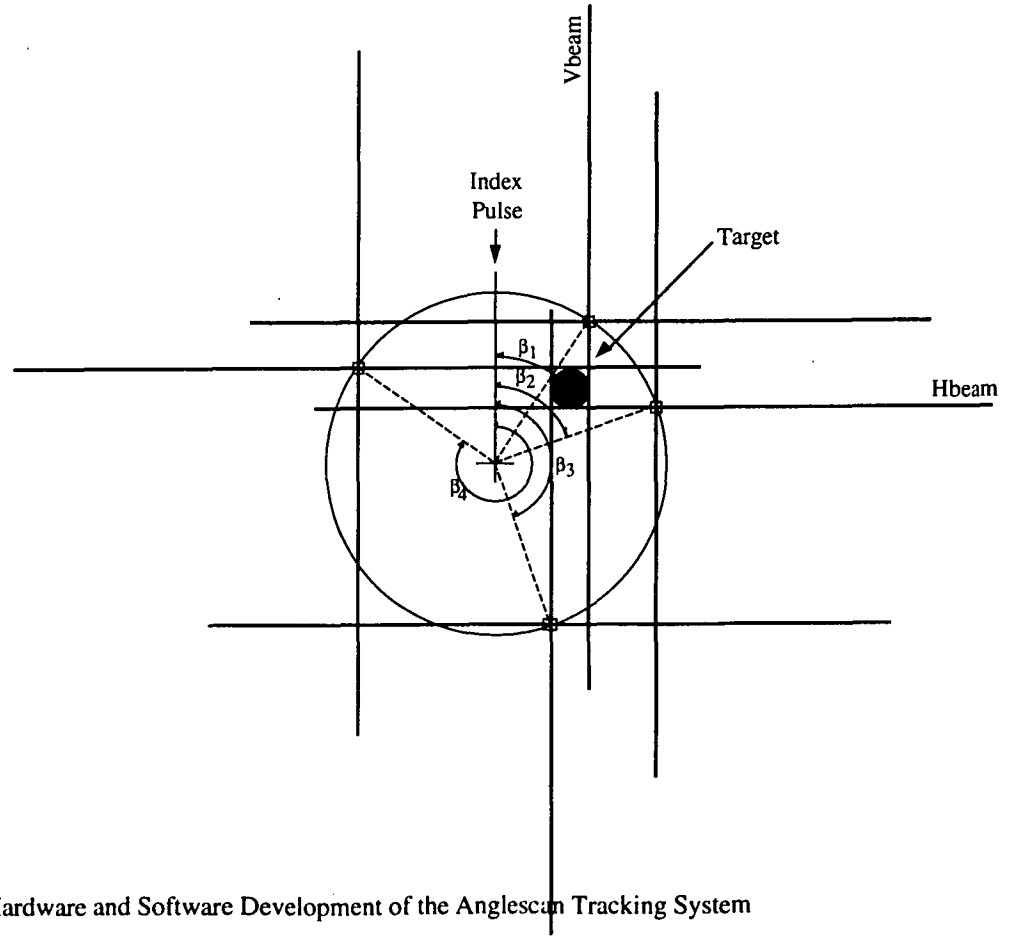
A copy of the circuit diagram is provided in appendix 7.5B. Each of the four-16 bit counters consists of two cascaded 8 bit binary counters (8 x 74HC590) which have internal octal latches and tri-state bus drivers.

Interface to the backplane bus is via a 40 pin IDC connector marked J1 shown on the right of the diagram. Address decoding logic and selection of each counter is with a decoder (74HC154). Gating control logic is with a 8 bit serial-in parallel-out shift register (74HC164). Overcount and undercount conditions are detected and stored as bit flags by a separate octal buffer (74HC245).

There are two 10 pin IDC connectors marked J2 and J3. The rotary encoder on the rotating wedge prism is connected to J2. The return laser signal pulses are fed to connector J3.


gating control logic
laser
A
index
B
encoder pulses
C
enable
clock
clear
α_1 counter
α_2 counter
α_3 counter
α_4 counter
store
16-bit latch
OE
OE=output enable
bus
cpu
Address decoding logic
title: Fig.2.5A Target Measurement Card
size: A4
rev:
file:
date:
sheet of:
drn by:


Vbeam
Index
Pulse
Target
Hbeam
α1
α2
α3
α4


Vbeam
Index
Pulse
Target
Hbeam
β1
β2
β3
β4

[2.5] Azimuth and Elevation Interface Cards:

These cards directly interface between the microprocessor and the azimuth and elevation servo-motors. Both motors are rated at 24V and 500mA at maximum torque.
The interface performs the following tasks: (1) To generate PWM signals to actuate the servo-motors, (2) Coarse positioning counters from the rotary encoders on the servo-motors giving a positioning resolution of 0.01 degrees per step, (3) Fine positioning counters giving a positioning resolution of 0.001 degrees per step.
Due to the complexity, the servomotor interface boards have being separated into two identical boards, one for the azimuth and the other for the elevation servomotor. Each board has different jumpers to give a different I/O address map for selection by the microprocessor, chapter 7.7B has the full I/O address map.
The design is open loop (refer to figure.2.6A) with the microprocessor closing the control loop making different controller implementations possible in software. To implement a control system, the microprocessor loads a 16 bit binary number into the PWM registers which determines the duty cycle. To obtain positioning information, the microprocessor reads a 16 bit number from the positioning counter which is the displacement of the output shaft (ie: the gearbox output).

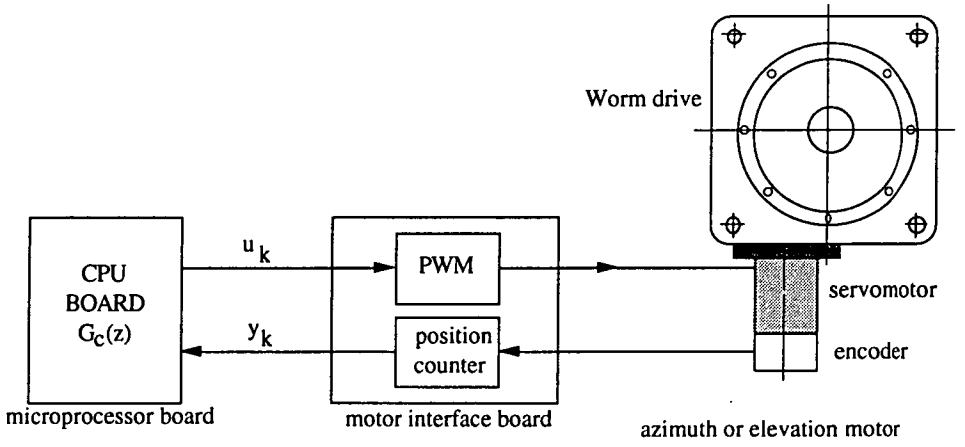


Fig.2.6A Azimuth and Elevation Servosystem

In the next chapter, system identification techniques will be used to determine the transfer function of the above system: Y(z)/U(z) for both the azimuth and elevation servosystems. The system transfer function will later be used for the design of an appropriate control system. The original design consisted of a dedicated off the shelf motion controller IC, the LM628 which had all the internal circuitry to fully implement a digital PID controller. We found this design to be unsuitable for our application and consequently redesigned the azimuth and elevation boards with discrete logic. In the next section we briefly discuss why the LM628 design was unsatisfactory for this application.

The LM628 Digital PID Controller:

The LM628 motion controller IC (manufactured by National Semiconductors) is a dedicated controller which fully implements a 32 bit digital PID compensator using integer arithmetic, it also has internal 32 bit quadrature counters for interfacing to positioning rotary encoders. Additionally, it has an internal trapezoidal velocity profile generator and and antiwindup integration limiter. Figure.2.6 shows the block diagram

The LM628 internal registers are fully under microprocessor control. The control registers set up the velocity profile, slew rate, displacement, and PID filter coefficients. The microprocessor can also read the LM628 internal 32 bit quadrature counters for positioning measurements.
The general cascaded inclusion of the LM628 into a control loop is illustrated in fig 2.6A below:

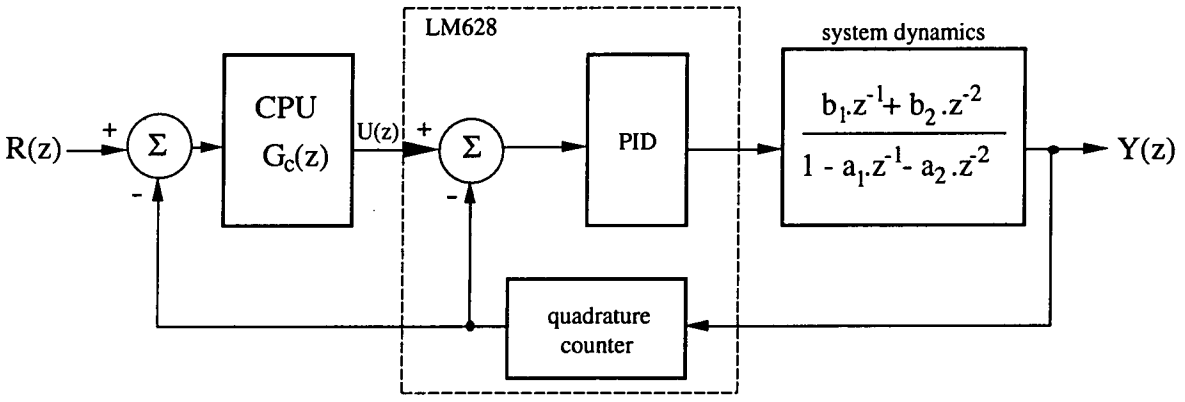


Fig.2.6B LM628 PID controller

The intergrator antiwindup limiter prevents the integration term from building up to excessively large values which causes actuator saturation. The output of the LM628 is a DC voltage which drives a power amplifier and the servomotor.
Furthermore, the LM628 is fully microprocessor compatible, and interfaces to a microprocessor data and address bus easily. The circuit diagram of the LM628 design is shown in appendix 7.5E.

Initially we designed and constructed a prototype azimuth and interface motor controller using this IC, but it was found to be too restrictive in implemention (PID only) and not therefore suitable for implementing more appropriate controller structures for tracking applications. However, some of the advantages of PID control are listed below.

Advantages of PID control:

[a] Zero steady state error due to a pole located at the origin (integral action).
[b] Can operate at lower gains resulting in more stable systems.
[c] The PID controller has been found to be a reliable and robust industrial controller.

The transfer function of a PID controller in the s domain has the structure:

$$G_{PID}(s) = K_p + \frac{K_i}{s} + K_d.s \qquad (2.2)$$

K_p=proportional gain term, K_i=integration term, and K_d=difference term. A PID controller is also more commonly known as a 3 term controller. The DC drive method shown in Fig.2.6 was found to have two distinct disadvantages which make the design impractical for our application:

[1] Servomotor systems driven with a DC voltage across the armature have higher inherent stiction and deadband compared with PWM systems.

[2] The system shown in Fig.2.6 has two feedback loops, the combination of the two loops places the closed loop poles of the system in the left hand side of the z plane within the unit circle creating an undersampling condition. Finally, the PID structure show in Fig.2.6 is fixed, does not allow the design of a controller structure $G_C(z)$ which best suits the plant, the internal loop is very fast (300 usec) compared to the slower outer loop to the processor (30 msec).

This design is unsuitable for our application requiring the tracking of moving targets, its best application is found in static positioning, for example: XY plotters, NC drilling and milling machines.
The original controller was based on the LM628 design. A copy of the circuit diagram is included in the appendix section 7.5E. Notice that the design incorporates both the azimuth and the elevation drivers on the same board. One of the advantages of this design was extreme simplicity because the LM628 contained all the internal control and counter logic to directly interface to a servomotor and to the microprocessor circuit.

The main disadvantage however was the complexity in the software required to setup and control the LM628 with all the internal registers and configuration options available. This makes the interface program cumbersome and difficult to properly test.

Furthermore, the controller structure and different sampling rate made the control system design and simulation complicated.
It was decided to adopt a PWM design, this was fully implemented in discrete logic. The concept of this design is that the microprocessor simply loads a 16 bit binary number into the interface card. This number represents the duty cycle of the applied PWM signal to the motor, denoted by u_k. A 16 bit binary counter is used as feedback to measure the actual displacement., refer to figure.2.7. The PWM drive system is further described below.

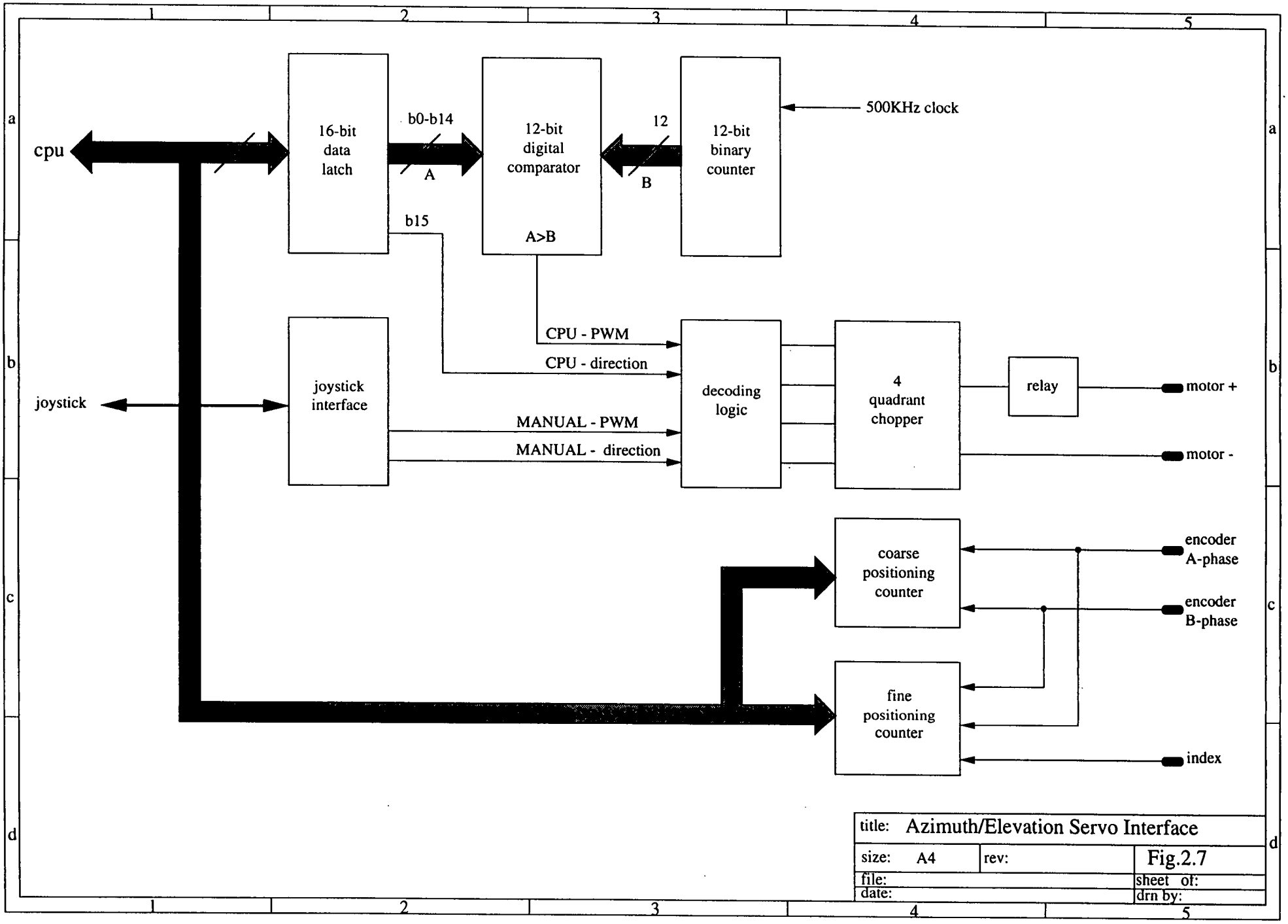
The PWM Drive System:

The schematic diagram of a Pulse Width Modulation (PWM drive) system is shown in figure.2.7. The microprocessor loads a 2 byte word which determines the length of the pulse. Only the lower 12 bits are used, this gives the pulse width a duty cycle adjustable from 1/4096 to 4095/4096. Bit 15 is used to determine the direction of rotation, (b15=0: forward, b15=1: reverse). Bits b12,b13,b14 remain unused.
This procedure provides a control signal to the motor in the range of -4095 to +4095 in steps of 1.

Refering to Fig.2.7: the heavy shaded lines indicate the bus backplane connections. The microprocessor writes a 16 bit word to a 16 bit data latch. A digital comparator compares the output of a 12 stage binary counter with the stored valued on the data latch providing a high pulse when input (A) is greater than the counter (B).

The decoding logic stage selects either one of two inputs: (1) CPU; or (2) MANUAL. This enables both automatic tracking under microprocessor control or manual positioning control respectively. Manual positioning control uses a joystick interface to initially position the tracking system on the desired target. The decoding logic selects one pair of inputs consisting of a **PWM duty cycle** and **PWM direction** and generates four outputs to drive a four quadrant chopper ie: four FET transistors in an H configuration, a standard off the shelf IC: L293E manufactured by SGS-Thompson Microelectronics is used as an H driver, this has the required current and voltage ratings to directly drive the servomotor.

One of the advantages of using a four quadrant chopper driver is in efficiency, the transistors are either fully on or fully off. In either case the power dissipation by the transistors is minimal. This is ideal for battery operated portable instruments.


cpu
16-bit data latch
b0-b14
A
12-bit digital comparator
A>B
12
B
12-bit binary counter
500KHz clock
b15
joystick
joystick interface
CPU - PWM
CPU - direction
MANUAL - PWM
MANUAL - direction
decoding logic
4 quadrant chopper
relay
motor +
motor -
coarse positioning counter
fine positioning counter
encoder A-phase
encoder B-phase
index
title: Azimuth/Elevation Servo Interface
size: A4
rev:
Fig.2.7
file:
sheet of:
date:
drn by:

The motor rotary encoder produces two quadrature output signals: A-phase and B-phase. Each step indicates a 0.01 degree displacement. The COARSE POSITIONING COUNTER decodes the two signals and produces coarse displacement measurements in steps of 0.01 degree resolution. A standard 16 bit quadrature counter IC manufactured by HP is used (HCTL2020) which contains all the internal counter counter logic, and is microprocessor compatible.

In addition, a separate counter: FINE POSITIONING COUNTER is used to provide positioning measurements accurate to about ten times better than the coarse positioning counter, ie to 0.001 degree resolution. The fine positioning counter is able to measure fractions of a step from the rotary encoder. Figure 2.10 illustrates the actual implementation.

PWM Frequency and Pulse Width Selection:

The criteria for selecting the PWM frequency and minimun pulse width is based upon two time constants: (1) motor electrical time constant, (2) mechanical time constant. The two parameters have been estimated from the inertia, friction coefficient, motor inductance and resistance parameters supplied by the manufacturer.

Electrical time constant: The minimum pulse width T_W must be greater than the electrical time constant of the motor, otherwise the average armature current will have insufficient time to build up, thus:

$$T_w(\min) > \frac{La}{Ra} \qquad (2.3)$$

where La=armature inductance, Ra=armature resistance. Given that La/Ra=50 usec, then the minimun pulse width should be of the same magnitude, thus T_W = 50 usec.
Note that pulses below this figure maybe sufficient to allow the armature current to build up and overcome deadband and stiction.

Mechanical time constant: The mechanical time constant determines the PWM frequency $1/T_p$. If the frequency is chosen below $1/T_m$, where T_m is the mechanical time constant, then the motor will appear to jump or step each time a pulse is applied.
Thus:

$$T_p(\max) < T_m \qquad (2.4)$$

given that T_m=1/70, then T_p(max) < 0.014 or 14 msec. The two values above provide limits to the design of the pulse width modulator: see figure.2.8 below.

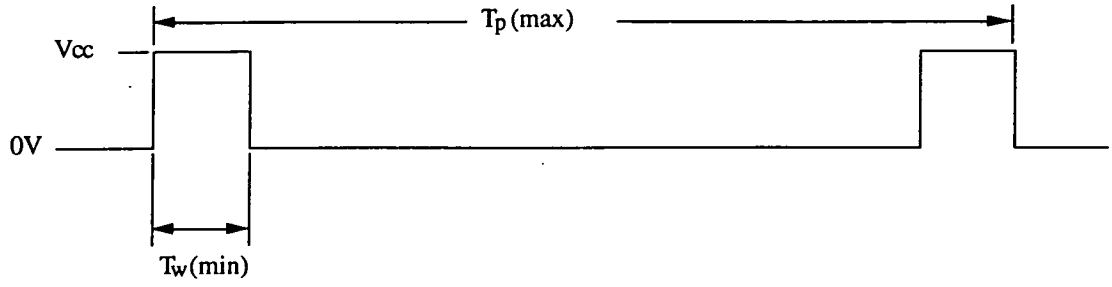


Fig.2.8 PWM Minimum Pulse and Maximum Period

Fine Positioning Measurements:

The fine positioning counter described earlier in this section allows a greater positioning resolution to be obtained. The coarse positioning counter only resolves position to a single step from the optical rotary encoder, this is 1/100 of a degree. The fine positioning counter has one disadvantage however, that is the requirement that the servomotor is rotating. It is otherwise unable to provide fine measurements.
To obtain greater positioning resolution, the fine positioning counter is capable of measuring fractions of a pulse step or subpulse.

Refering to figure 2.9, consider the two waveforms: the first is a typical output from the rotary encoder, because the output constists of two signals 90 degrees apart (Phase-A and Phase-B) then the waveform shown represents the EXCLUSIVE-OR of the two signals. Note that the separation between C_k and C_{k+1} is one encoder step resolution ie: 0.01 degrees.
The waveform below is a typical return pulse signal showing the four measurements of $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$ with respect to the index pulse previously described in section 2.4.
Suppose that the coarse positioning counter measures the count C_k just prior to the index pulse occouring, this is an exact integer count, and this value is stored.

The fine positioning counter restarts on each transition (of the rotary encoder pulse) and stores the value of 'n' upon the detection of the index pulse. It also measures the lenght of the pulse 'N' and stores it. Figure 2.9 illustrates these parameters.

We now have two measurements: a coarse measurement C_k indicating integer steps of the motor rotary encoder, and a fine measurement n/N indicating fractions of a step.

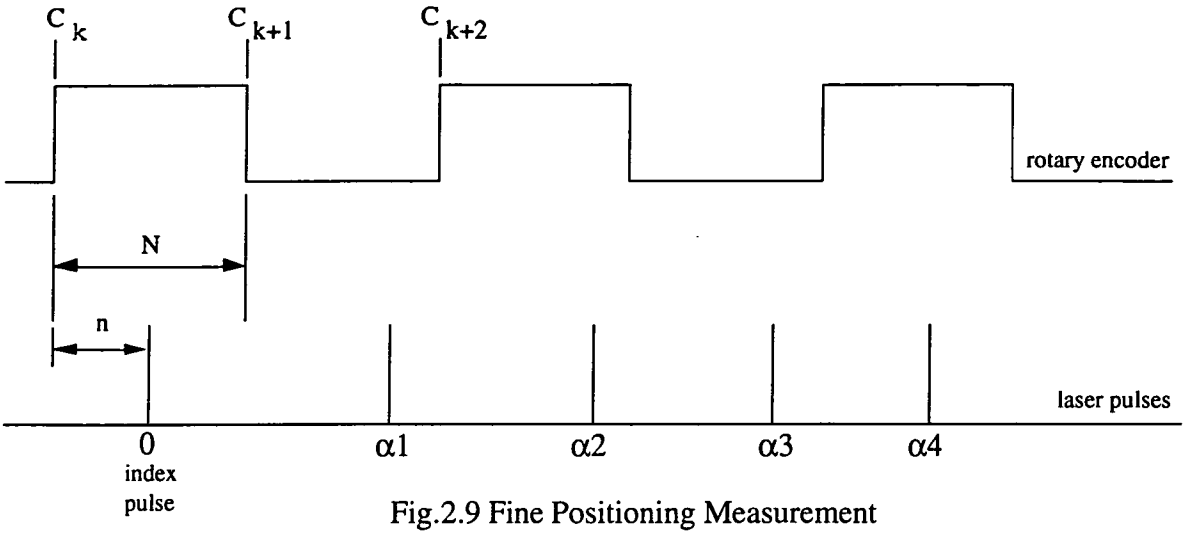


Fig.2.9 Fine Positioning Measurement

The position of the target can therefore be computed in the following way:

$$\text{X-position} = C_k + \frac{n}{N} + \text{azimuth} \qquad (2.5A)$$

$$\text{Y-position} = C_k + \frac{n}{N} + \text{elevation} \qquad (2.5B)$$

Where **azimuth** and **elevation** are calculated using equations (1.8A) and (1.8B). Note also that both n, N and C_k used in equations (2.5A) represent measurements from the azimuth board, whereas in (2.5B) represent measurements from the elevation board.

The term C_k+n/N is the displacement measurement up to the point where the index pulse occours, thereafter the calculation of displacement using $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$ data is referenced to the index pulse.

Figure 2.10 shows the fine positioning measurement logic, a common clocking signal is fed into 2 separate 8 bit binary counters. Both counters n and N reset and restart upon the occourence of each edge (either trailing or falling) from the rotary encoder as shown in figure 2.9 for example when the C_k edge occours both counters simultaneously reset.

If an index pulse is detected, the instantaneous counter values are latched into two octal tri-state drivers. The values of n and N can be read at any time. However at the occourence of the next index pulse these values will be updated (every 30 msec). The software waits for the index pulse prior to reading the counter outputs.

Figure 2.10 below illustrates the basic design of the fine positioning counter. The complete circuit diagram is shown in the appendix section 7.5C.

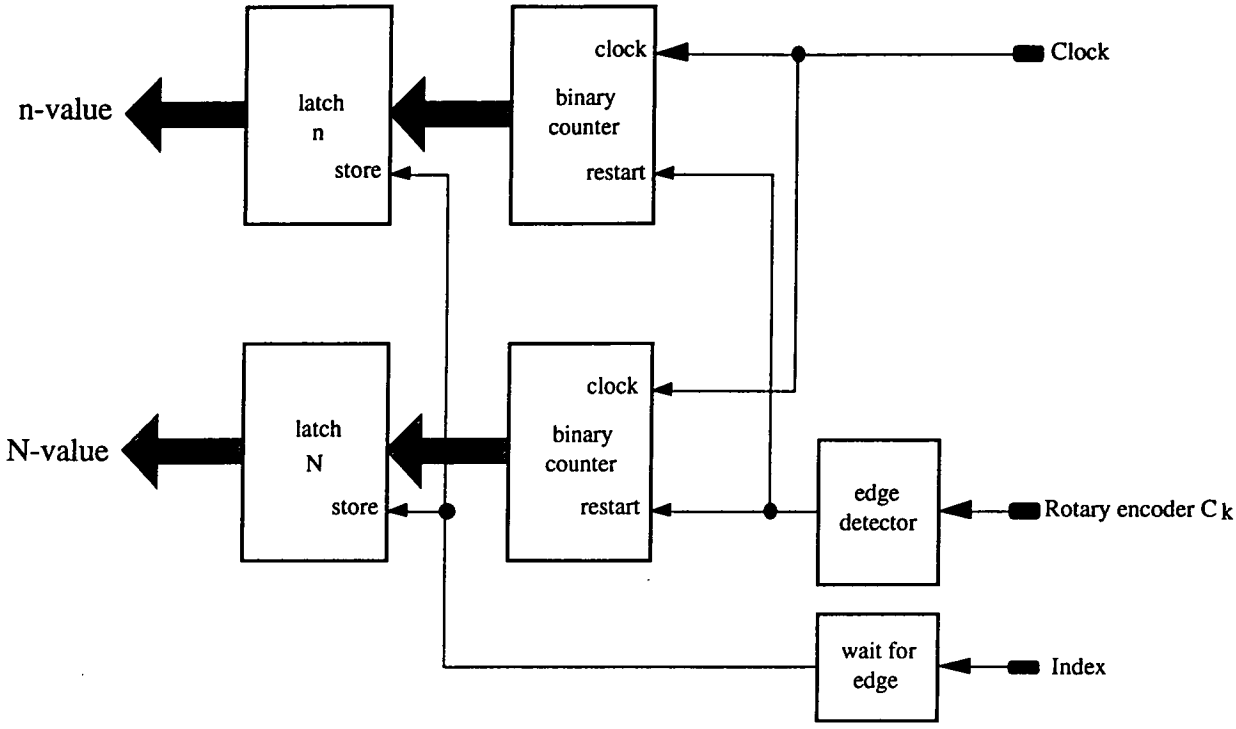


Fig.2.10 Fine positioning measurement logic

Figure 2.10B below is the L293E, a four quadrant chopper motor driver integrated circuit which uses FETs rather than transistors. The decoding logic is external to the IC package. This design offers low power consumption by the drive stage.

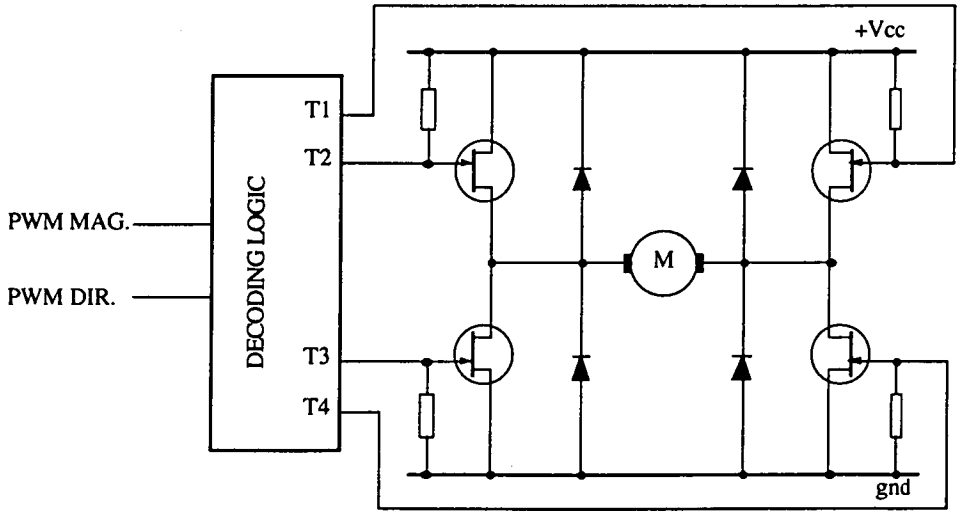


Fig.2.10B Typical Four Quadrant PWM Chopper Configuration.

Circuit Diagram:

A copy of the circuit diagram is given in the appendix 7.5C. The 40 pin connector (top left) is to interface with the backplane bus, IC2 and IC3 latch the 16 bit PWM word u_k. The digital comparators (IC3,4,5) compare the data from a binary counter (IC6) with the input value of u_k (also refer to figure 2.7) producing the pulse width output. The pulse width counter uses IC16,17 (measures N, see equation.2.3A,B), the pulse fraction counter uses IC13,14 (measures n). The coarse positioning counter uses a dedicated quadrature counter IC12 (HCTL2020, measures C_k). Address decoding is with IC10, and the servomotor driver uses IC18 (L293E).

[2.6] Analog Signal Processor Card:

The analog signal processor card inputs the raw return laser pulse signals to generate a rectangular TTL compatible signal output. The return laser pulse signals occur each time the vertical or horizontal beams of the cross intercept the target mirror.
The purpose of the signal processing card is to produce a reliable and accurate TTL signal output regardless of the input signal amplitude (target distance or size), DC offsets (due to ambient light), and stray signals (from reflective objects within the field of view).
From measurements, the signal to noise ratio at a distance of 700 metres is sufficient in amplitude to allow detection by pre-filtering (low pass) the signal and then applying threshold detection. This is the method used.
Recall that for each rotation of the laser cross, four return pulses are generated if the target is within the field of view of the instrument. The return pulses may not all be of equal amplitude, this is due to the optical transmission system which cannot be guaranteed to produce identical intensity along each axis. Figure 2.11 illustrates a typical return laser pulse sequence.

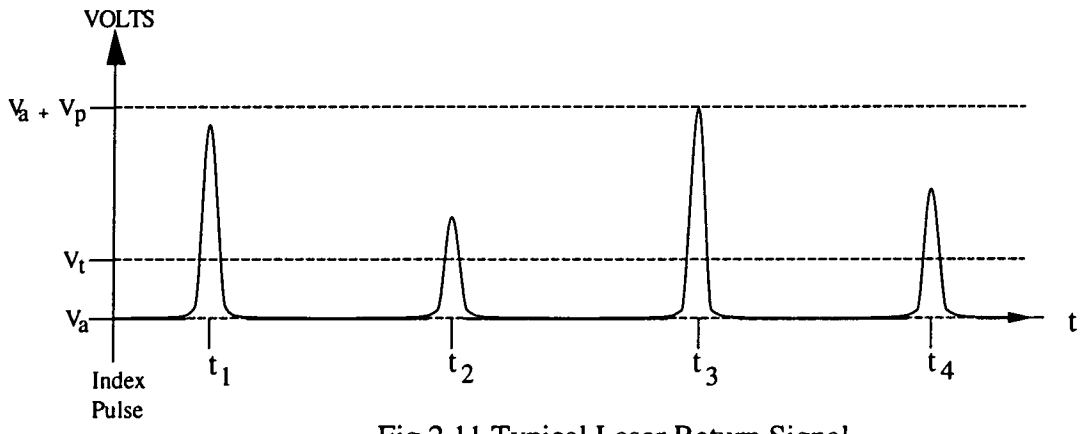


Fig.2.11 Typical Laser Return Signal

Refering to figure 2.11, the three parameters which are required for signal detection include:

Va: Offset voltage due to background ambient light.
Vt: Threshold detection logic.
Vp: Signal peak amplitude

Typical values of signal peak to peak amplitude V_p are given below as a function of range:

Range (m)	Vp (mV)	Va(mV)
7	500	200
700	5	200

The DC offset due to ambient background light can be several hundred millivolts in level. Subsequently at a range of 700 metres, the signal is only 5 mV superimposed on several hundred millivolts DC offset component. As mentioned earlier, not all return pulses in the same rotation are equal in amplitude, the difference is caused by the intensity variation between the vertical and horizontal beams transmitted. This can produce pulses up to 3 times in amplitude difference within the same rotation or data sequence. We looked at two similar techniques for producing the required TTL level output signal:

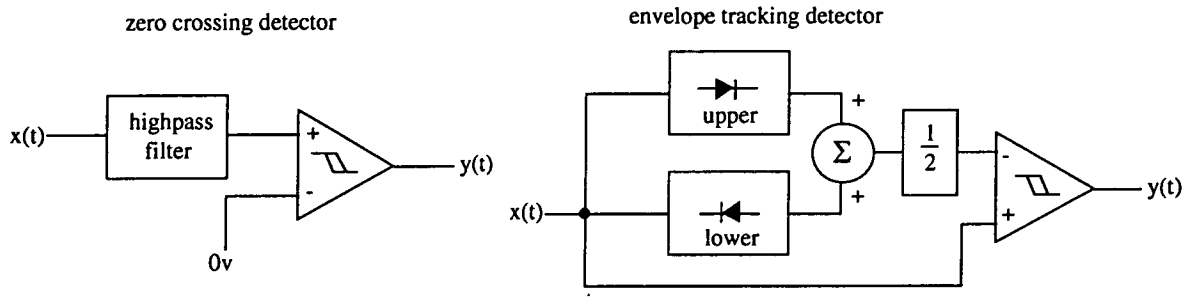


Fig.2.12 Two methods of signal detection

[1] **Zero Crossing Detector:** AC couple the signal to remove the DC component, then apply a zero crossing detector. AC coupling uses a high pass filter, this is illustrated in figure.2.12. The method has the disadvantage of having a slow response to changing ambient light level (eg: the instrument FOV may be passing through a bright region while tracking), additionally AC coupling distorts the shape of the signal.

[2] **Envelope Tracking Detector:** Detect the upper envelope voltage and lower envelope voltage, then set the threshold detector to the mean of the two values as shown in Fig.2.12b below. This method is satisfactory if all the pulses have the same amplitude.

A variation of the second method has being used. Instead of using the outputs of the upper and lower envelope tracking detectors as inputs to the schmitt trigger, the two values are read by the microprocessor, the threshold is then calculated in software and loaded into an 8 bit D/A converter which determines the threshold on the inverting input of the schmitt trigger stage. A block diagram of the analog signal processing card is illustrated in fig.2.14. The signal is initially amplified and low pass filtered to remove high frequency noise and provide a signal to the first stage ambient detector and subtractor to remove any DC offset due to background light.

The next stage is a programmable gain amplifier controlled by the microprocessor, this is required due to the dynamic range of the signal (amplitude). Next both the signal amplitude and any residual offset are measured by the microprocessor and the appropriate threshold voltage is placed into the DA converter. Lastly a digital de-glitching filter is used to remove any narrow glitches occouring during slow rise / fall times of the input return laser signal. Figure.2.14 shows a block diagram of the overall system consisting of the following stages:

[1] **First Stage LPF + Gain:** This is an active low pass filter, second order butterworth (fc=150KHz) and amplifier with DC gain=10. Even at maximum input signal amplitude of 0.5Vpp, it does not cause the amplifier to saturate The filter attenuates high frequency noise.

[2] **First Stage Ambient Detector:** Detects the amount of ambient background light using a baseline (most negative) envelope detector. It produces the measurement of Va (see Fig.2.13D).

[3] **Subtractor:** This stage removes the DC offset Va from the waveform (see Fig.2.11) thus clamping the signal baseline to 0V.

[4] **Programmable Gain Stage:** The gain is selectable from 1 to 64 in steps of 1. Control input is a 6 bit binary word specifying the gain. It also includes a second active butterworth filter (second order) with a cutoff frequency fc=300KHz. (Fig.3.13B).

[5] **Threshold Detector:** A high speed threshold detector with 5% hysteresis. The threshold level Vt is software selectable from 0.00V to 2.55VDC.

[6] **Digital Deglitching Filter:** It eliminates glitches by sampling the signal with a 3 stage D-type F/F, then ANDING the sampled signals together, (see Fig.2.13).

[7] **Signal Offset:** Measures any residual signal offset (signal baseline) Va. This should be near zero.

[8] **Signal Peak:** Envelope detector, measures the signal maximum amplitude Vp. It provides an indication of the signal amplitude in which the microprocessor subsequently decides upon setting the threshold detection level in **5** above. See Fig.2.13C.

[9] 8-Bit Data Latch: Octal D-type latch for storing the programmable gain.

[10] 8-Bit A/D: Microprocessor compatible A/D converter with input analog multiplexer.

[11] 8-Bit D/A: Microprocessor compatible D/A converter to set threshold voltage Vt. The threshold value is set according to the equation:

$$V_t = \frac{1}{4} V_p + V_a \tag{2.6}$$

[12] Gain Switches: Analog CMOS switches, for switching in different resistor combinations from a feedback resistor network.

The circuit diagrams below illustrate the actual implementation of some of the stages by the signal processing card, fig.2.13 is a simplified digital de-glitching filter, Fig.2.13b is a programmable gain stage amplifier controlled by switching feedback resistors using CMOS analog switches, and Fig.2.13c shows signal level and baseline detectors.

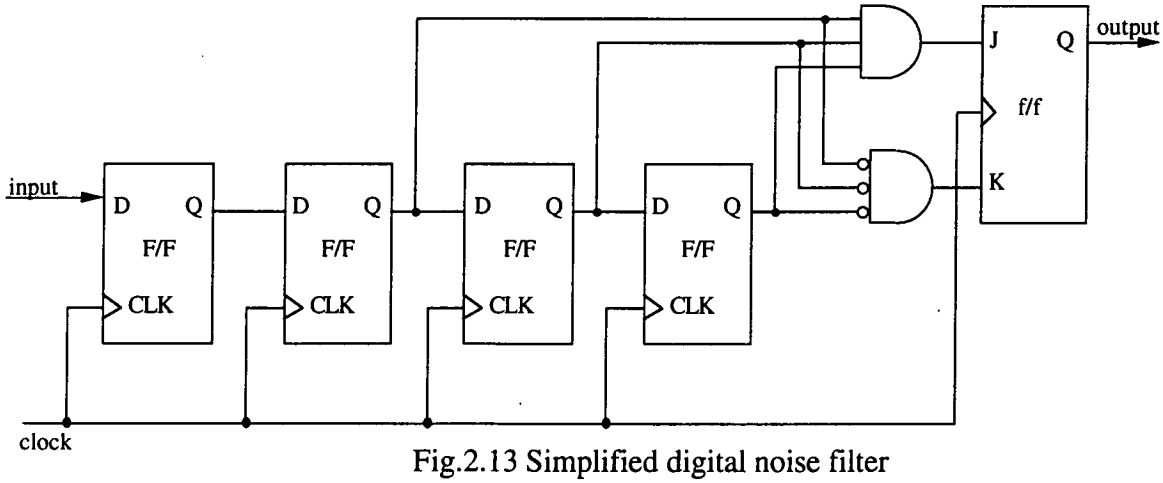


Fig.2.13 Simplified digital noise filter

Figure 2.13b illustrates the basic design of the programmable gain stage. Analog CMOS switches are used to switch in different feedback divider circuits, nominal on resistance of about 300 ohms for analog swiches.

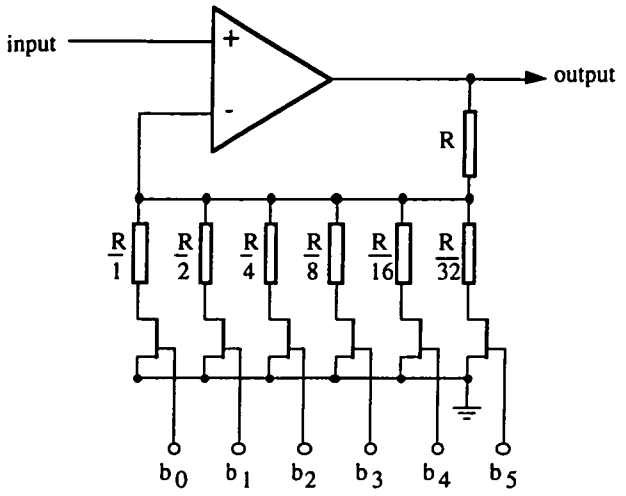


Fig.2.13b programmable gain stage

The programmable gain amplifier can select gains from x1 to x63 in steps of 1. The gain is given by the following relation:

$$GAIN = 1 + \sum_{j=0}^{5} 2^j.b_j \qquad (2.7)$$

The following circuits implement the upper (V_p) peak signal detector. The baseline ambient detector (V_a) is identical but with the direction of the diode reversed. The RC time-constant in the circuit is to gradually discharge the capacitor after the last peak has occured. The design is that of an envelope detector.

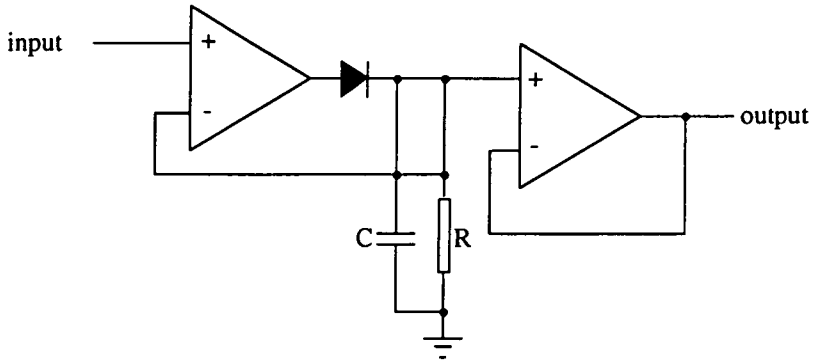
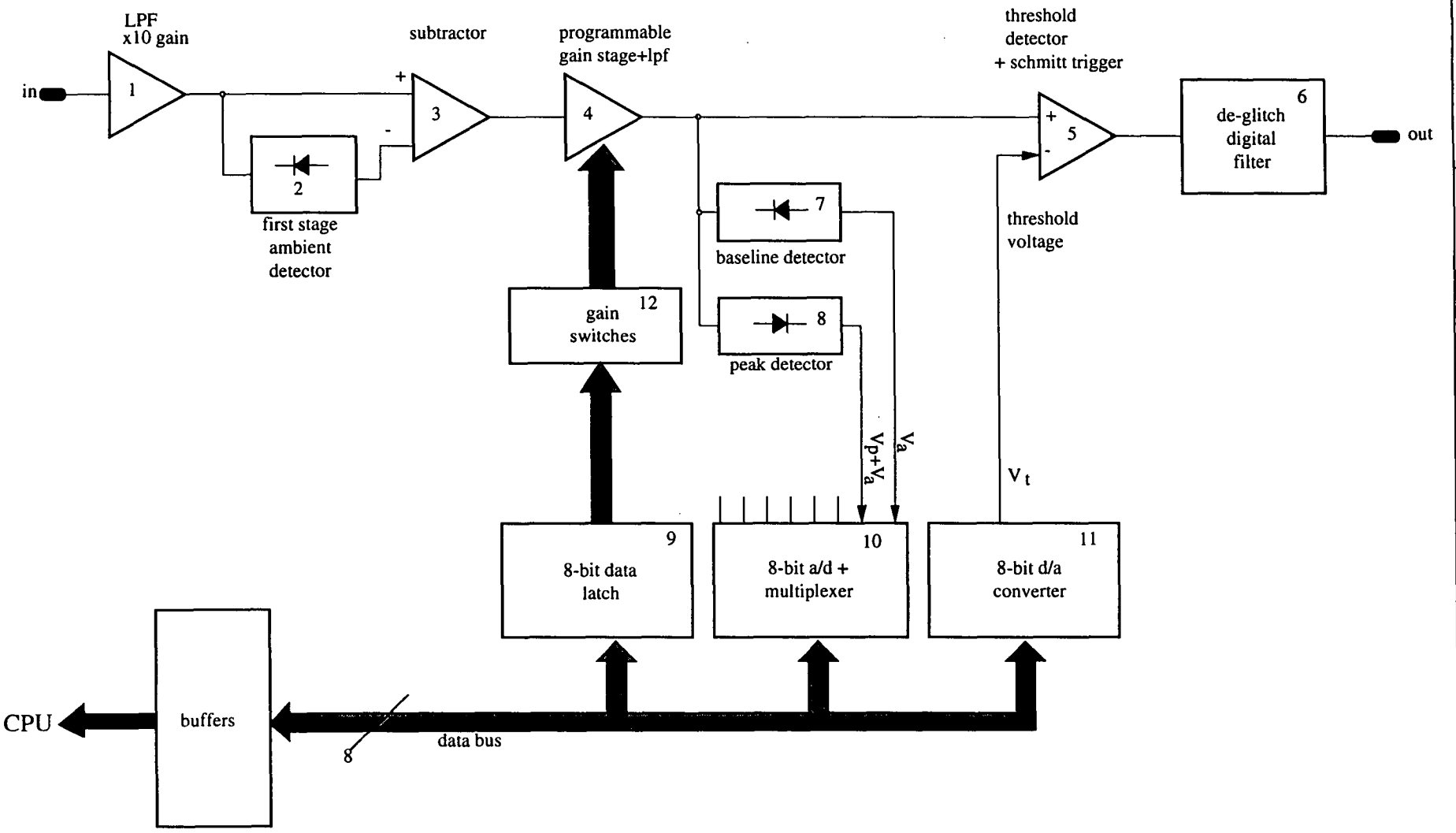


Fig.2.13c peak detector stage

Fig.2.14 on the following page illustrates the overall block diagram of the analog signal processing card, a full schematic diagram is also supplied in appendix 7.5A.



title: Analog signal processor board.			
size: a4	rev:	Fig.2.14	
file:		sheet of:	
date:		drn by:	

[2.7] Microprocessor Card:

The microprocessor board was originally designed by Mr. C. Ashworth (Physics Department, University of Tasmania) several years earlier. A functional block diagram of the microprocessor system is shown in Fig.2.15. The microprocessor card has the following specifications:

Z80 CPU running a 4.0MHz clock
32K bytes of RAM memory (max).
32K bytes of EPROM memory (max).
Z80 PIO general purpose peripheral I/O with two 8 bit ports: A and B.
Z80 SIO dual serial RS232 communications port (DART).
RS232 level translators and line drivers.

The development of the software system was carried out on an IBM-PC environment which includes: text editors, a HITECH-Z80 C cross compiler which is a standard high level ANSI C cross compiler for Z80 target systems, and an EPROM programmer connected to the PC. The advantage of using a high level C compiler is that it enables complex floating point arithmetic to be emulated in software, including trigonometric functions. It also allows faster development time and a more robust software system.

The microprocessor card has up to 32K of EPROM memory which is sufficient to allow the entire C program (about 100 pages of code and comments) to be loaded.
The software has been entirely coded in C, no assembly language programming was required. This is primarily due to dedicated hardware designed such that no critical timing or real time response was required by the microprocessor. (A general overview of the program is given in section 2.9).

Refering to figure.2.15, the I/O port designated PORT-A and PORT-B consists of two 8-bit fully programmable data buses. The two ports interface to the backplane bus (see Fig.2.3 above). Ports are allocated in the following way:

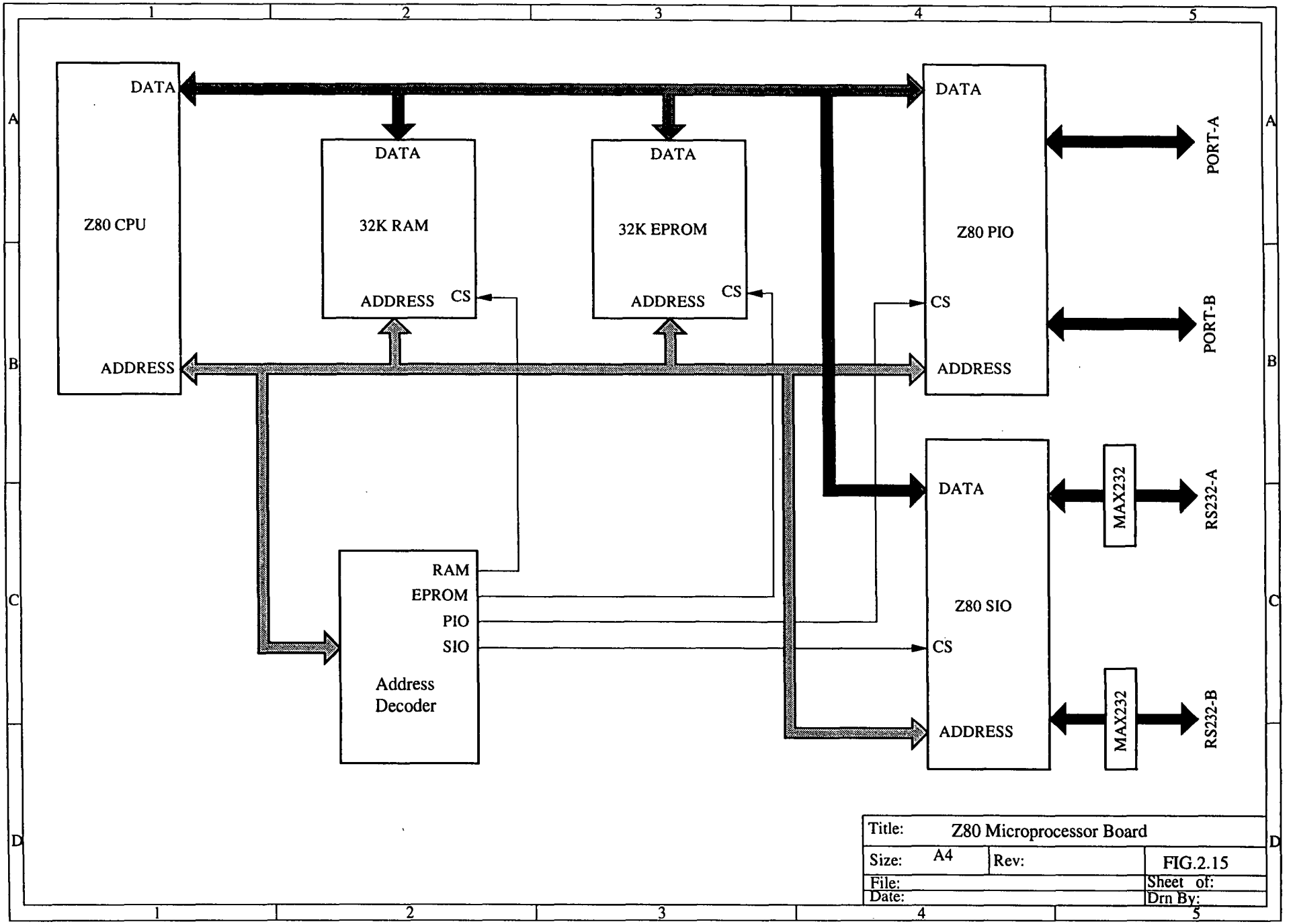
Port-A: is used as a bidirectional general purpose data bus to read and write to peripheral boards and individual registers (buffers) on individual boards.

Port-B: is used as an 8 bit device select bus (address bus) to select different peripheral boards and individual addresses on each board.

PORT-B is only 8 bits wide, address decoding will only allow up to 256 different I/O addresses (or peripheral registers) to be accessed. A full description of the I/O address map is given is section 7.7B.

Two serial RS232 ports are available on the microprocessor board, CHANNEL-A: is used to communicate with a host terminal, a description is given in section 7.7A on the user interface. The second channel: CHANNEL-B serves to dump data or status to a serial printer, but is currently unused. Serial communication drivers have being written in C. All the serial communication software is found in the file ATSDART.C which contains all the necessary interface (routines) for reading and writing to the two serial RS232 ports.

The Z80 board was found to be inadequate for the application as computational requirements placed a severe overload on the processor, it is therefore recomended to eventually move to a full 32bit processor with math coprocessing on board. This is necessary for proper implementation of on board auto-calibration routines, tracking algorithms and closed loop controller implementations.


DATA
Z80 CPU
ADDRESS
DATA
32K RAM
ADDRESS
CS
DATA
32K EPROM
ADDRESS
CS
DATA
Z80 PIO
CS
ADDRESS
PORT-A
PORT-B
DATA
Z80 SIO
CS
ADDRESS
MAX232
MAX232
RS232-A
RS232-B
RAM
EPROM
PIO
SIO
Address
Decoder
Title: Z80 Microprocessor Board
Size: A4
Rev:
FIG.2.15
File:
Sheet of:
Date:
Drn By:

[2.8] Front Panel Display Card:

The front panel card was originally built for the purpose of having a fully portable instrument for field use. Currently however it remains unused, apart from a few simple error messages which are displayed on the LCD. We prefer to use a terminal (or a portable PC for field use) and communicate directly with the microprocessor via the serial port.

Figure 2.15b on the following page shows a functional block diagram of the front panel display and keyboard card, it also contains a bus-backplane connector to allow direct interface with the microprocessor.

We have also mounted some DB25 and DB15 connectors directly on the front panel printed circuit artwork such that the connectors protrude through the slotted front panel aluminium face plate. The purpose of having front panel mounted connectors is to simplify connecting and disconnecting all cabling to the anglescan unit.

Connections:

There are two DB15 connectors, one labelled **count** which is connected to the rotary optical encoder mounted on the rotating wedge motor (refer to figure 2.1). The second DB15 connector labelled **pulses** is connected to the photodetector which receives laser return pulses (refer to figure 2.2).
Additionally, there are two DB25 connectors. The first is labelled **servomotors**, this is connected to the azimuth and elevation servomotors and rotary encoders. The second DB25 is labelled **RS232** and is connected to the serial port of a terminal or console.

LCD Display Drivers:

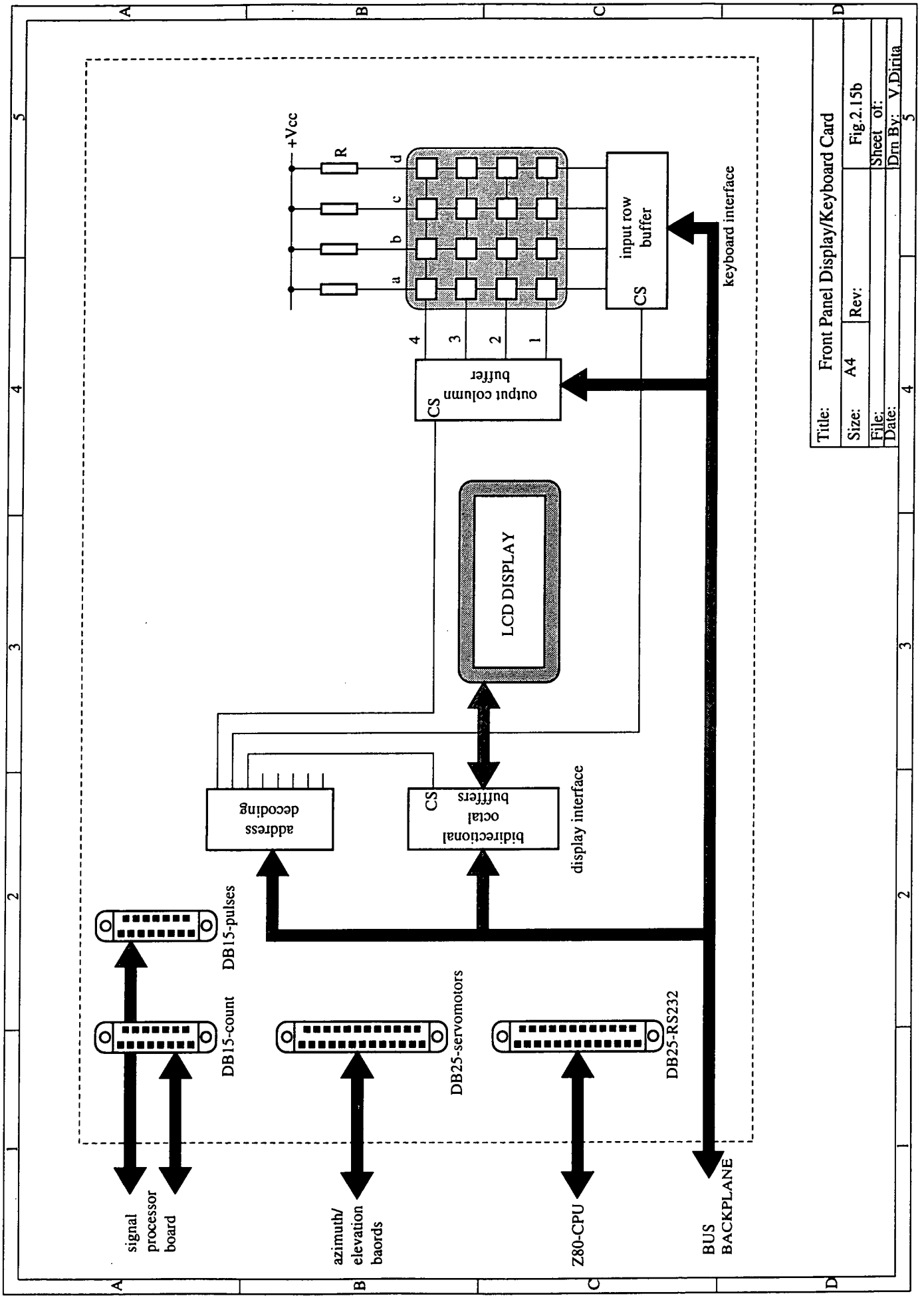
The LCD display interface is fully microprocessor compatible with the exception of a bidirectional octal tri-state buffer on the data port. The LCD contains internal data and control registers to select differing modes of display.
Software interfacing to the LCD consists of a C source file: **ATSDISP.C**, which has the necessary drivers to control and display the LCD. The two most low-level display drivers are found in this file and are called: **DisplayGetChar()** and **DisplayPutChar()** which read and write characters from the LCD display buffer.

Keyboard Interface:

Keyboard scanning is done entirely in software without the need of dedicated keyboard decoder ICs. The keyboard has 16 hexadecimal keys. Currently this keyboard is unused. The following routines are found in the c source file **ATSKPAD.C** which has all the necessary keyboard interfacing routines.
The two routines which provide information about the status of the keyboard are: **kpadavail()** which returns a true/false condition depending if a key is currently being pressed, and **kpadread()** which returns the character being pressed.
Note that the keypad interface is not interrupt driven, instead the microprocessor polls the keyboard periodically testing for a key pressed condition. If true then the key is subsequenly read in.

The keyboard scanning routine **kpadread()** determines the key being pressed by using 2 four bit buffers labelled 1,2,3,4 and a,b,c,d as shown in figure 2.15b. To determine which key is pressed, the program places 0111 (binary) on the output column buffer and scans the input row buffer. If a key is pressed in the first row then one of the four bits is zero depending on the key. Otherwise it continues with placing 1011, 1101, 1110, testing in each instance for any zeros occuring in the input row buffer. The position of the zero in the output column buffer and the position of the zero in the input column buffer determines the row and colum of the key being pressed.

The circuit diagram is shown in the appendix 7.5D.


+Vcc
R
d
c
b
a
4
3
2
1
input row buffer
CS
keyboard interface
output column buffer
CS
LCD DISPLAY
address decoding
CS
bidirectional octal buffers
display interface
DB15-pulses
DB15-count
DB25-servomotors
DB25-RS232
signal processor board
azimuth/ elevation baords
Z80-CPU
BUS BACKPLANE
Title: Front Panel Display/Keyboard Card
Size: A4
Rev:
Fig.2.15b
Sheet of:
File:
Date:
Drn By: V.Dirita

[2.9] Anglescan Software: Description:

The anglescan software is written in high level C language, consisting of approximately 100 pages of souce code (and comments) compiled with a HITECH-Z80 cross compiler. The compiler runs on an IBM-PC, and the resulting binary file requires about 32K of EPROM target memory to load.
All the development work is carried out on a PC, this includes text editors, C-cross compilers, source level debuggers and EPROM programmer connected to the PC. The software is designed to operate the instrument in three modes.

Modes:

[1] CALIBRATION MODE: Enables the instrument to be calibrated on site, section 5 provides a mathematical background of calibration. (Currently not implemented).
[2] TRACKING MODE: Which is the main mode of operation after the instrument has been calibrated. It also includes target search, aquisition mode. (Partly implemented).
[3] DEBUGGING MODE: This is the current mode of operation which it enables both hardware and software development to be tested. It also includes a simpler form of target tracking and calibration algorithms. (Fully implemented).

Program Structure:

The software has been partitioned into separate C source files, each file contains an individual group of functionally related subroutines, this design methodology allows the program to retain consistency and structure. During compilation, the files are included in the main program and individually recompiled. We have arranged the files in the following groups:

ATSMAIN.C: Contains the three operating modes: Calibration, Tracking and Debugging modes.

ATSDEFN.H: Contains the hardware definition, memory map, I/O map, global constants and static variables.

ATSDART.C: Contains all subroutines for Dual Uart RS232 serial communication interface drivers

ATSKPAD.C: Contains all subroutines for interfacing to the front panel keypad, (the keypad is currently not in use).

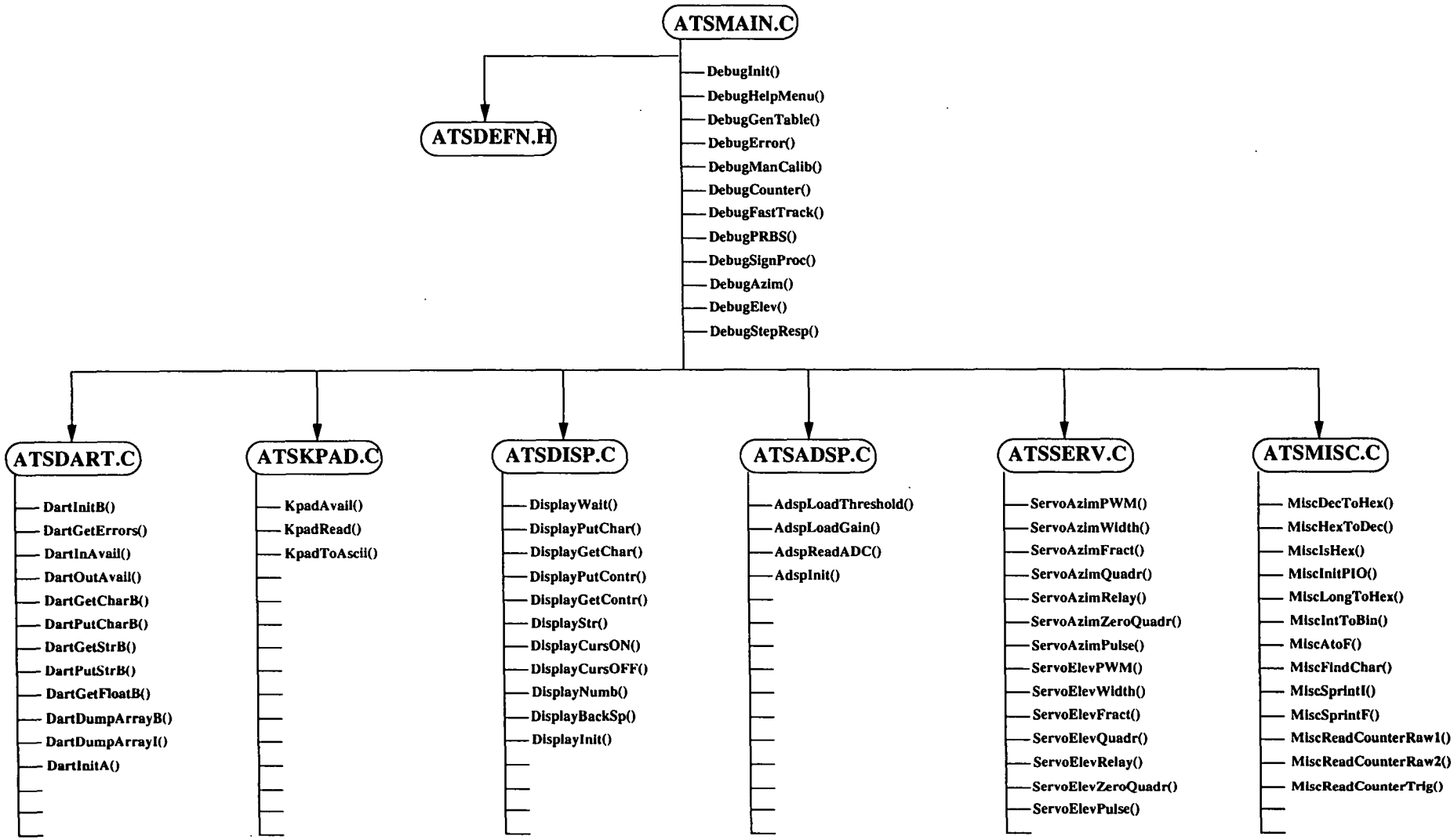
ATSDISP.C: Contains all subroutines for interfacing to the front panel LCD display. The LCD is currently only used to display the "power-up ok" message after turning the instrument on).

ATSADSP.C: Contains all subroutines for interfacing to the analog signal processor card described in section 2.6 above.

ATSSERV.C: Contains all subroutines for interfacing to the azimuth and elevation servo motors described in section 2.5.

ATSMISC.C: Contains all other (miscellaneous) subroutines. Loading the C standard library "stdlib.c" requires substantial EPROM memory because the library is loaded in its entirety. Rather than using the C library, the required routines were re-written in C.

Figure.2.16 illustrates the organization of the program, all filenames are shown inside rounded-rectangular boxes, all the functions below each box refer to individual subroutines contained in that particular file. The program is structured in two levels: Level-1 is the file 'ATSMAIN.C', Level-2 are all the other files below. One basic rule about subroutine calling is that routines in the higher level can call routines on the lower levels, but not the other way. Furthermore, recursive routines are not permitted This allows easier debugging and also improves the robustness of the program. The advantages of using C include faster development time to produce working software and the availability of standard floating point arithmetic and trigonometric functions implemented in IEEE standard floating point format 32 bit accuracy.



Title:	Software Decomposition	
Size: A4	Rev:	FIG.2.16
File:		Sheet of:
Date:		Drn By:

[2.10] User Menu Interface

Curently the software operates in debugging mode only. When the instrument is initially turned on with the serial port connected to a dumb terminal (any VT100 emulation), the terminal will display an on screen menu of options. One of these options also includes an auto-tracking algorithm, a calibration algorithm, measurement and hardware debug options. The on-screen-menu is illustrated in figure 2.17:

COMMAND	DESCRIPTION
H	List the Help Menu (this menu)
C	Calibrate the Instrument
D 1/2/b	Read Counter #1/#2/b=both
T	Fast Tracking Algorithm+Limit Filters
P a/e	PRBS Response on a=azimuth / e=elevation
I t=x.xx/g=xxx/a	g=xxx gain t=x.xx threshold a=ADC data
S a/e=xxxx	Step Response a=azimuth e=elevation
A p/r=xxxx/q/w/f	Azimuth p=xxxx(PWM) q=quadrature w=width f=fraction.
E p/r=xxxx/q/w/f	Elevation p=xxxx(PWM) q=quadrature w=width f=fraction.

Fig.2.17 On screen menu after power up

The commands are fully described in detail in chapter 7.7A. Some of the more important ones are briefly described below:

C: This command invokes a simple instrument calibration procedure. (Chapter 5 describes this calibration technique). This procedure is capable of solving only for two of the five unknown calibration coefficients: (1) Index pulse Ip, and (2) field of view R.

T Implements a closed loop tracking system, this can either be a simple proportional controller, or a more complex controller such as a pole-placement or digital PID structure. Currently only implementing a proportional controller.

S [a/e] This option does a simple transient step response on the servosystem. It is used primarily for model verification in chapter-3. Two options are a=xxxx is a step response on the azimuth servosystem with amplitude xxxx hundredth's degrees, similarily e=xxxx is for the elevation servosystem.

P [a/e] This option implements a PRBS test for system identification purposes, the microprocessor applies a random binary signal u(k) to the motor driver board and measures the angular displacement $\theta(k)$, at the end of the test which generally lasts for approximately 10 seconds, the data pairs u(k) and $\theta(k)$ is dumped to the host terminal into a file.

We use this option in chapter-3, the options a=azimuth servo, and e=elevation servo, allow the test to be performed on each servo-system individually.

Target Determination:

The C language implementation of floating point arithmetic allows the computation of trigonometric functions which are necessary for both calibration and target tracking. However because of the lack of a floating point co-processor, all the floating point arithmetic is emmulated in software by the C cross-complier. Floating point emulation results in slow computation time, about 10 msec to calculate **sin(x)** or **cos(x)**. For fast target tracking this is too slow, and rather than computing the trigonometric functions directly, we initially create a look-up table. Each entry to the table is equivalent to calculating the function: **sin(x)**.

The table contains 1000 entries ranging from 0 to 90 degrees. Additionally a straight line interpolation scheme is used between adjacent points. Straight line interpolation between two adjacent points is calculated by looking at the following figure.2.18:

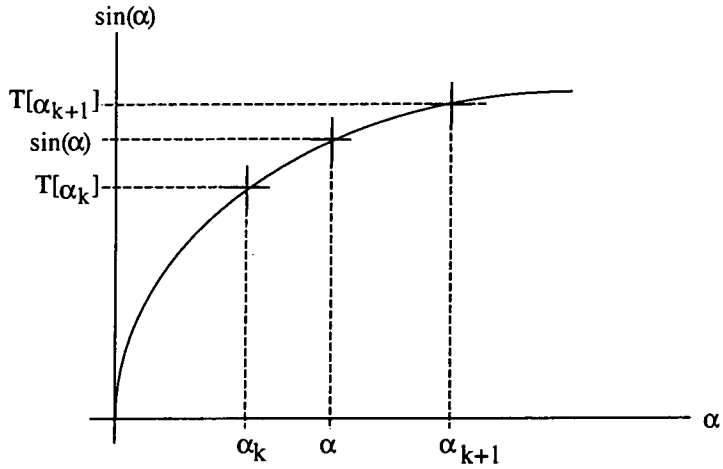


Fig.2.18 Linear Interpolation Scheme

If T[ ] is a look up table with entries at discrete values of the angle α, containing the sine of the angle thus: $\sin(\alpha_k) = T[\alpha_k]$. Then to determine the sine of any angle α, use the following straight line interpolation equation:

$$SIN(\alpha) = T[\alpha_k] + \frac{T[\alpha_{k+1}] - T[\alpha_k]}{\alpha_{k+1} - \alpha_k}(\alpha - \alpha_k) \qquad (2.8)$$

Where $\alpha_k < \alpha < \alpha_{k+1}$.

Note that this table can also be used to compute cosines necessary by target determination equations (see equations 1.6 and 1.7). This table remains resident in memory.

[2.11] Chapter Summary

In this chapter, a brief outline of the Anglescan Tracking system was presented. The hardware was discussed in functionality only. However detailed circuit diagrams have being included in the appendix together with photographs for reference. The Anglescan instrument software was also discussed with reference to its general structure and user interface.

Five boards including the PCB overlay for the anglescan electronic system have being designed, constructed and tested: [1] Front panel display and keyboard card, [2] PWM Servo-motor interface card of which two have being built (azimuth and elevation) [3] Analog signal processing card, [4] Target measurement card of which two have being built, [5] and the original PID servo-motor interface card which has both azimuth and elevation drivers. The original PID servo-motor interface is no longer used.

The software was also briefly covered. A listing of the original source code is not included in the thesis but we have provided a functional overview and decomposition highlighting the structure of the program and its features. Its user interface menu has changed constantly over the duration of the project, initially the menu only provided simple hardware debugging routines. These were progressively phased out and have being replaced by more complex menu features such as tracking, calibration and identification algorithms. The user interface is described fully in the appendix 7.7A.

The program was developed on a PC and compiled to a binary file which can be loaded directly into EPROM. The EPROM programmer card plugs directly into an IBM-PC. The cross-compiler used was a Z80 Hitech-C which was a low cost compiler but implemented the full ANSI C language with additional features specific to the processor architecture such as vectored interrupt handling. All the software was developed fully in the C language, without any need to use assembler language routines.

The program occupies the entire 32K of EPROM memory, this being the maximum memory available on the microprocessor card. It was found that this was a restriction to prevent further development of the software such as more complex control algorithms and implementation of a full calibration procedure.
Another restriction with the Z80 microprocessor card was the speed of floating point computations. Both of these problems will need to be addressed if further development of the project is to continue in this area.

Chapter-3

System Identification:

[3.1] Introduction:

Identification as described by[J.P.Norton [3.13] is defined as: *"The process of constructing a mathematical model of a dynamical system from observations and prior knowledge"*, Other definitions of identification by Zadeh [3.15] include: *"Identification is the determination, on the basis of input and output, of a system within a specified class of systems, to which the system under test is equivalent"*.
For our objectives, the purpose of identification is to obtain a model of the azimuth and elevation servomotor systems which will then enable us to design a controller to improve the tracking performance of the anglescan tracking system.
An accurate model may not be a realistic aim because of the variability in the system parameters. Errors such as disturbances, measurement inaccuracies and limitations in the model structure all contribute to the uncertainty of the model.

Models may be specifically determined for standard input functions such as unit step, pseudo-binary-random-signal (PRBS), sinuosoid or specific input disturbances, or may be determined for the general case where a control system model and simulation is required for any type of input. The choice of test signals depends on the bandwidth of the system being identified, furthermore the input test signal should contain sufficient and even excitation at all frequencies to cover the full spectrum of the system which is being identified. Note also that we deal only with systems that are assumed to be linear, time invariant and causal. An excellent introduction to system identification is given by Astrom and Eykhoff [ref. 3.14]

[3.2] Identification Techniques:

A number of identification techniques exist, some of the more specific ones are described in the following literature by: L.Jung [3.5], P.Eykhoff [3.7], K.J.Hunt [3.8], R.Stanway [3.9], and J.P.Norton [3.13].

Identification generally involves a process of iteration which is repeated to the stage where the model structure describes the actual system to a desired accuracy. This can be determined from step response tests and compared to simulation results. The model structure may have to be changed a number of times and possibly increase the order of the model until it meets the design criteria based on some error funtion. We could also have on line identification for adaptive control purposes.

A detailed survey study of system identification can be found by a paper from K.J.Astrom, "System Identification, A survey" [3.14]. Briefly, system identification falls into one of three categories listed below:

(1) Classical methods:
(2) Correlation methods:
(3) Regressive methods:

The classification is not distinct and clear-cut, the distinction is characterized by two types of models: (1) non-parametric models: in which the classical methods and correlation methods apply, it has the advantage that it is not necessary to specify the order of the process explicitly. (2) Parametric models in which regressive methods apply, in this instance we solve for the coefficients of a known model structure. It has the disadvantage of giving large errors if the model structure does not agree with the order of the process.

Other methods also include stochastic least squares, Bayes estimation, and deconvolution. A brief description of the classical, correlation and regressive methods is outlined below.

(1) Classical Methods:

Classical methods deal with known input functions u(t) which are applied to the unknown system, and simply measuring the output y(t) in time domain. Generally, a step input or an impulse function is used. Impulse functions are commonly approximated by a narrow pulse width T_p and amplitude $1/T_p$.

The output response can be found by convolving the input function u(t) with the system transfer function g(t). The convolution function in continuous time is described by the following equation 3.2:

$$y(t) = \int_{\tau=0}^{t} g(\tau).u(t-\tau).d\tau \qquad (3.2)$$

If $u(\tau)$ describes a pulse function of height$=1/T_p$ and width T_p, the system transfer function in the time domain is g(t), then the output y(t) approaches the system transfer function g(t) as the limit of T_p approaches zero and the input function approaches the impulse function $u(t)=\delta(t)$

(2) Correlation Techniques:

Correlation techniques deal with averaging to reduce errors due to the presence of noise and measurement errors. Correlation techniques require the determination of both the discrete time cross correlation function $R_{uy}(k)$ and autocorrelation function $R_{uu}(k)$.

$$R_{uy}(k) = \lim_{N\to\infty} \frac{1}{2N+1} \sum_{i=-N}^{N} u_i.y_{i+k} \tag{3.4A}$$

$$R_{uu}(k) = \lim_{N\to\infty} \frac{1}{2N+1} \sum_{i=-N}^{N} u_i.u_{i+k} \tag{3.4B}$$

Where u_k and y_k represent the values of the input and the output at time t=kT, where T is the period between samples and k is the k'th sample of the continuous time variable u(t) and y(t), ie: u_k=u(kT), and y_k=y(kT).
The discrete time convolution function described in equation 3.2 above can be written as:

$$y_k = \sum_{j=0}^{N} g_j.u_{k-j} \tag{3.5}$$

Denoting j'th sampled value of the continuous time variable g(t) by g_j. Taking cross-correlation of each side with respect to u in discrete time:

$$R_{uy}(k) = \sum_{j=0}^{N} g_j.R_{uu}(k-j) \tag{3.6}$$

To find the function g_j, some appropriate test signal (u_k) with known auto-correlation function is required such as white noise. Discrete time white noise has a flat spectrum, it has the disadvantage of causing saturation if large values are applied however. Alternatively deterministic long period number sequence such as PRBS, usually m-sequence maximal length with period=2^m-1 where the autocorrelation function is also periodic and is known.

When selecting a PRBS signal, the bit interval should be less than the fastest time constant of the system to be identified, and when selecting the period or duration of the test, it should be longer than the settling time (of the impulse response) or longest time constant.

Generating a PRBS signal in a high level language (eg. C) is relatively simple, it can otherwise be generated in hardware with the use of a shift register as shown in Fig.3.1 below:

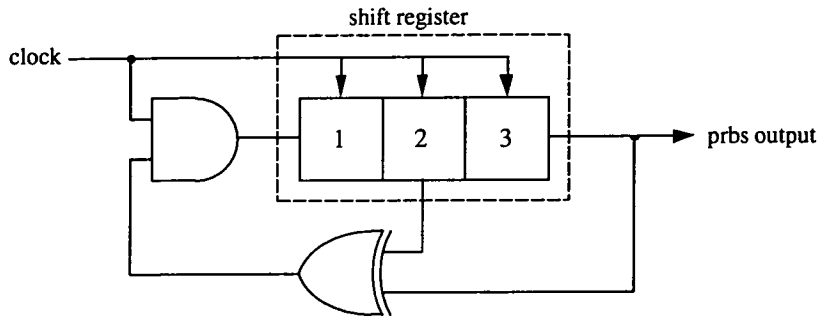


Fig.3.1: 7 Bit m-Sequence Shift Register

Equation 3.6 represents a discrete time convolution function, then by taking discrete Fourier transforms of each side of equation 3.6, the convolution becomes a product in the frequency domain. This is refered to power spectral density, written in frequency domain form:

$$G(j\omega) = \frac{P_{uy}(j\omega)}{P_{uu}(j\omega)} \tag{3.8}$$

where:

$G(j\omega)$ is the frequency response of the system

$P_{uu}(j\omega)$ is the discrete fourier transform of $R_{uu}(\tau)$

$P_{uy}(j\omega)$ is the discrete fourier transform of $R_{uy}(\tau)$

The main disadvantage is that this method only provides the frequency response of the system, and does not give us the unkown parameters of the model. This method will not be uses for identification.

(3) Linear Regressive and Least Squares Methods:

Linear regressive methods use a large data set together with an assumed *a priori* structure of the model. It attempts to compute the coefficients of the model such that the predicted output matches the actual output with a minimum error. The simplest type has the form:

$$\underline{y} = \mathbf{H}.\underline{\theta} + \underline{e} \tag{3.9}$$

Where:

$\underline{y}=[y_1,y_2, \ldots y_M]^T$ is the measurement vector, for a set of M measurements.
$\underline{e}=[e_1,e_2, \ldots e_M]^T$ is the residual (error) vector associated with M measurements.
$\underline{\theta}=[\theta_1,\theta_2, \ldots \theta_N]^T$ is the unknown vector with N unknown model coefficients

and

$$\mathbf{H} = \begin{bmatrix} h_{11} & \cdots & h_{1N} \\ \cdot & & \\ \cdot & & \\ h_{M1} & \cdots & h_{MN} \end{bmatrix}$$

H is the regression matrix representing a combination of input u_k and output y_k sequences representing the model structure. The individual elements of the matrix is not shown, but as an example equation 3.31 illustrates the typical entries of the regression matrix for a system with the known model structure given by equation 3.29.
The term $\underline{e}$ is the error or residual component which is to be minimized. The definition for the error function is a sum given as:

$$J = \sum_{k=1}^{M} e_k^2 \tag{3.10}$$

where the term e_k is the error associated with measurement sample k. Some of the advantages of recursive estimation include: on line parameter estimation of the plant, the system model can be dynamically updated based on observations up to the current time. This method is usefull for: (a) Self tuning controllers, (b) Matched filter tuning, and (c) Failure prediction. The solution to which the error function J is minimized results in the best estimate of the unknown vector:

$$\underline{\hat{\theta}} = \left[\mathbf{H}^T\mathbf{H}\right]^{-1}\mathbf{H}^T.\underline{Y} \tag{3.10B}$$

The above method of least squares is non recursive and produces a result with all the measurements having equal weight. This method of non recursive least squares will be applied in section 3.5.

Alternatively, a recursive estimation technique is available in which updated estimates of the unknown model coefficients $\underline{\theta}$ is calculated each time a new set of measurements is obtained. This has the added advantage of being adaptive, and can be used in adaptive control, adaptive filtering, adaptive signal processing and on-line prediction. The two common methods of recursive identification (refer to K.J.Hunt, 3.8) are Recursive Least Squares and Extended Recursive Least Squares.

In the equations which follow, the polynomial terms $A(q^{-1})$ and $B(q^{-1})$ refer to the unknown system transfer function using the delay operator q^{-1}, which is related to the parameter vector and the regression vector.

Recursive Least Squares (RLS): The recursive function takes the more general form for each measurement at k'th sample:

$$A(q^{-1}).y_k = B(q^{-1}).u_k + e_k \tag{3.11}$$

Where: y_k is the k'th sample of the continuous time output $y_k=y(kT)$ at: t=kT
u_k is the k'th sample of the continuous time input $u_k=u(kT)$. at t=kT
e_k is the k'th residual or error term.
$A(q^{-1})=1 + a_1.q^{-1} + a_2.q^{-2} + ... a_n.q^{-n}$ represents a polynomial with: $a_1..a_n$ unknown coefficients.
$B(q^{-1})=b_0 + b_1.q^{-1} + b_2.q^{-2} + ... b_m.q^{-m}$ represents a polynomial with: $b_0...b_m$ unknown coefficients.

The model is illustrated by figure 3.3 below:

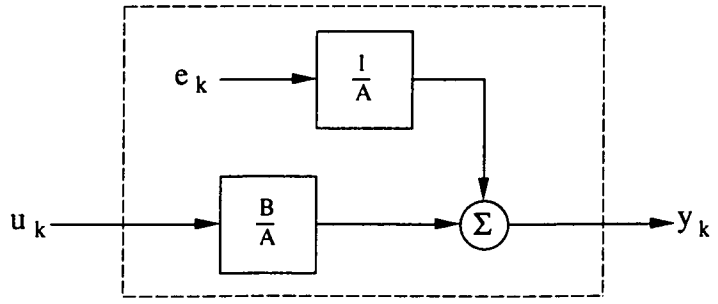


Fig.3.3 Model Structure

The error e_k is assumed to be serially uncorrelated and independent of the elements of the unknown vectors A and B. Identification using this method is applied to our servosystem model in section 3.6. There are a total of N=(m+1+n) unknowns. The model is illustrated by fig.3.3

Extended Least Squares (ELS): In cases where the noise term can be modelled as a moving average of a serially uncorrelated white noise sequence, we can write a recursive equation of the form:

$$A(q^{-1}) \cdot y_k = B(q^{-1}) \cdot u_k + C(q^{-1}) \cdot e_k \qquad (3.12)$$

Where $C(q^{-1})=1 + c_1.q^{-1} + c_2.q^{-2} + \ldots c_s.q^{-s}$ is a polynomial with c_1 to c_s unknown noise filter model coefficients.

The remaining terms are identical to equation 3.11.The model is depicted in fig.3.4. Section.3.7 uses this method of identification The terms $A(q^{-1})$, $B(q^{-1})$ and $C(q^{-1})$ describe filter polynomial functions of finite order. The method is similar to RLS, with the exception that it is extended by the inclusion of the disturbance term. The value of e_k is initially assumed zero ie $e_0=0$, it is then subsequently found by back substituting into the equation for the purpose of iteration:

$$e_k = y_k - \hat{y}_k \qquad (3.13)$$

Where y_k is the measured output, and y_k (hat) is the estimated output from the model coefficients. Each time the coefficients are updated, the estimated output and thus error term is calculated.

The product C(q).e(t) is assumed to be a correlated residual. There are now a total of N=(n+m+1+s) unknowns.

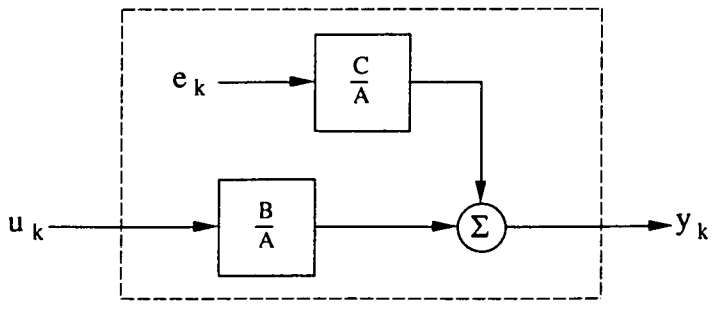


Fig.3.4 Model Structure

Transformation to the z domain from the delay operator q^{-1} notation is fairly straighfoward, thus from equation (3.11)

$$A(q^{-1}).y_k = B(q^{-1}).u_k$$

Where the polynomials $A(q^{-1})$ and $B(q^{-1})$ and y_k and u_k are as per equation (3.11). Taking z transform of the above expression and assume zero initial condition the following expression is obtained:

$$A(z).Y(z) = B(z).U(z)$$

where: $A(z)=1+a_1.z^{-1} + ... a_n.z^{-n}$

$B(z)=b_0 + b_1.z^{-1} + ... b_m.z^{-m}$

The transfer function is thus:

$$\frac{Y(z)}{U(z)} = G(z) = \frac{B(z)}{A(z)}$$

[3.3] The Servo System Model:

We can determine the general structure of the DC servomotor model in the z-domain. This will provide some indication to the structure and number of unknown model coefficients. Additionally, it also provides a relationship between polynomial coefficients and the motor servosystem physical constants such as Inertia, coefficient of friction, armature electrical time constants, back EMF coefficient and torque constant.
The following servosystem model ignores stiction, core/actuator saturation, deadband, and is assumed to be linear:

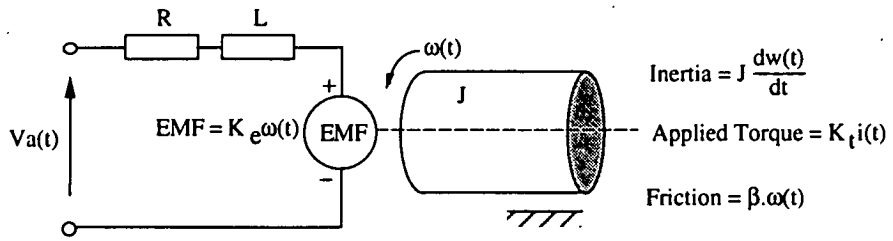


Fig.3.7 Armature Controlled DC Servomotor

Ignoring armature inductance L, the model in s-domain is shown in block diagram below. The input voltage Va(t) is the applied armature DC voltage:

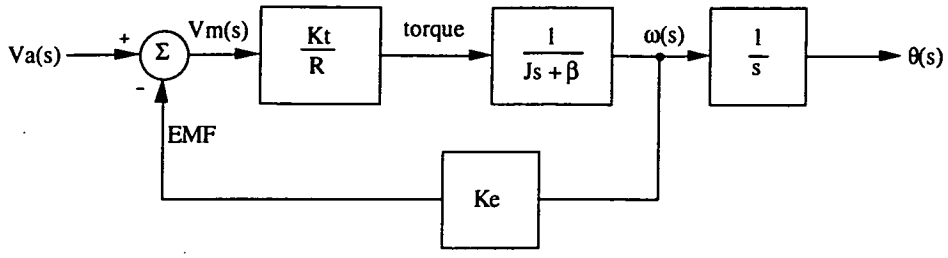


Fig.3.8 DC Servomotor Model

where: J=moment of inertia (motor+load), Kt=motor torque constant, R=armature resistance, β=coefficient of dynamic friction, Ke=back emf constant, θ(s)=ouput displacement in radians, and Va(s)=applied armature control voltage, ω(s)=output angular velocity (rad/sec.),

The closed loop transfer funtion H(z) evaluates to a second order system with two poles: one at the origin (first order system) and pole located at the mechanical time constant of the servosystem thus:

$$H(s) = \frac{\theta(s)}{Va(s)} \frac{\alpha_1}{s\,(s + \alpha_2)} \tag{3.24}$$

Where the terms are:

$$\alpha_1 = Kt / J.Ra$$

$$\alpha_2 = (\beta.Ra + Ke.Kt) / J.Ra \tag{3.25}$$

If armature inductance is included, the model would contain one additional pole and one zero. Taking z transform of equation (3.24) and including a Zero Order Sample and Hold (ZOH) due to the measurement system (the sampling time is set to Ts=30 msec) we obtain from z transform tables:

$$H(z) = \mathcal{Z}\left[\frac{1 - e^{-sT}}{s} . H(s)\right] \tag{3.26}$$

solving equation 3.26:

$$\boxed{H(z) = \frac{b_1.z + b_2}{(z - 1)\,(z - e^{-\alpha_2 T})}} \tag{3.27}$$

Where the b1,b2 numerator coefficients are given by equation 3.28 and the $\alpha 2$ is given by equation 3.25:

$$b_1 = \frac{\alpha_1}{\alpha_2^2}\left[\alpha_2.T - 1 + e^{-\alpha_2 T}\right]$$

$$b_2 = \frac{\alpha_1}{\alpha_2^2}\left[1 - e^{-\alpha_2 T} - \alpha_2.T.e^{-\alpha_2 T}\right] \qquad (3.28)$$

Note that when the pole has been determined in equation 3.27, it automatically fixes the values of b1 and b2 and hence the zero which is located by the ratio: b2/b1. There is effectively only one unknown parameter to be estimated, this being $\alpha 1$.

The complete control system has the structure illustrated in fig.3.9. Note that the controller designated as Gc(z) is implemented in software by the anglescan microprocessor. Additionally, system identification is also implemented by the same computer:

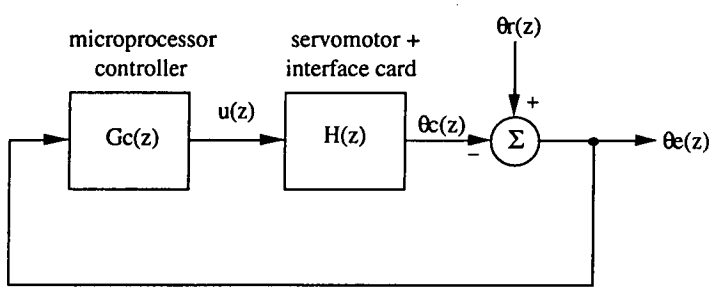


Fig.3.9 Control System Model

The function H(z) includes the cascaded combination of the servo-motor driver card (ie: azimuth or elevation interface card) and the transfer function of the servomotor system.

The controller (microprocessor) applies a test signal to the plant u(z), the test signal can either be a step function, or a PRBS random signal generated in software. We set the reference displacement to $\theta r(z)=0$, and then measure the output $\theta c(z)$.

The applied test signal u(z) is a 16 bit binary number which determines the duty cycle of the applied pulse width modulated signal. Thus a binary value of 1 gives 1/4096 duty cycle, and the uppermost value of 0FFF gives a duty cycle of 4095/4096 which is approximately 100%. Only the lower 12 bits of the binary number are used to determine the duty cycle. Bit b15 is used to determine the direction of rotation, thus 1FFF gives a duty cycle of 4095/4096 with the direction of rotation reversed.

The angular displacement is measured from the optical rotary encoder placed on the servomotor shaft, this value is measured in hundredths of degrees.

[3.4] Experimental Setup:

The experimental setup consists of the anglescan Z80 computer which has been programmed to apply an input test sequence (either STEP or RANDOM) u_k to the motor controller card, and to monitor the output response y_k. This information is stored during a test run. At the end of the test the information is downloaded via a serial RS232 interface from the anglescan computer to a host IBM-PC which runs the system identification programs.

Figure 3.10 below illustrates the setup and the model to identify, consisting of the Z80 microprocessor card and the servomotor driver card (either azimuth or elevation).

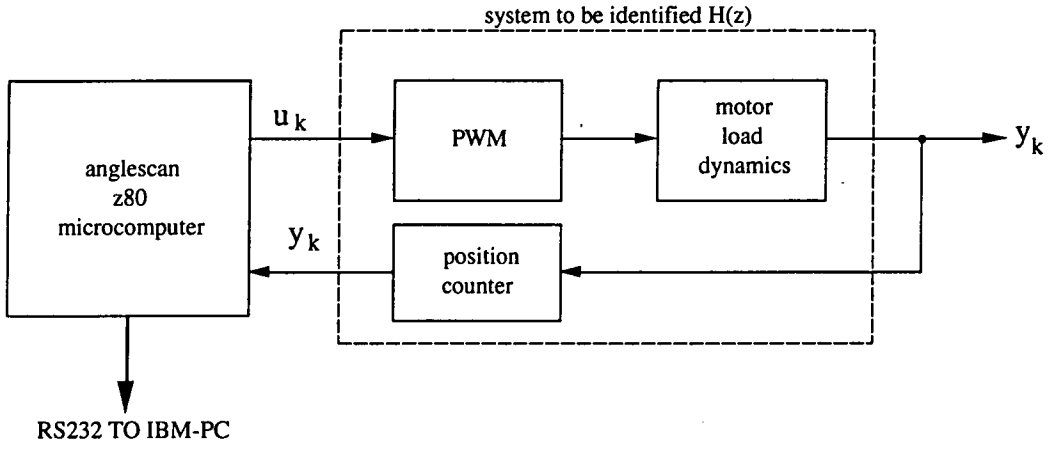


Fig.3.10 system identification, experimental setup

Note that the same setup will later be used to implement a closed loop controller with the anglescan Z80 microcomputer.

We denote the input test sequence by: $\{u_k\}$. The input test sequence is produced from a random number generator function (written in C code). The number is a binary value ranging from -4095 to +4095 (ie +/- 2^{12}). This number represents the duty cycle voltage applied to the motor. For example the number 1 generates a 1/4096 duty cycle as explained earlier. A negative value reverses the polarity on the motor. Subsequently the symbol $\{u_k\}$ refers to a 12 bit binary number.

The output sequence is denoted by $\{y_k\}$ and represents the motor displacement in hundredths of a degree. This unit is chosen because the resolution of the position rotary encoder on the motor resolves to 1/100 th of a degree. The output displacement sequence $\{y_k\}$ is thus a 16 bit binary number.

Measurements of y_k and outputs of u_k are made at sampling intervals of 30 msec. This value is chosen because the control system to be implemented later has a sampling interval of 30msec set by the mechanical rotation of the wedge prism. Note that the block marked "position counter" always indicates the true displacement y_k with no delay introduced. The values of $\{u_k\}$ and $\{y_k\}$ are stored in memory. Between 100 to 300 measurements are stored, samples are every 30 msec. The duration of the test is between 3 to 9 seconds which is considerably longer than the slowest time constant (mechanical time constant) or settling time.

Once measurements are completed, the anglescan computer downloads the information (u_k and y_k sequence) into an IBM-PC. The PC runs the identification programs (see appendix 7.4B) and determines the coefficients of the model transfer function H(z).

Conducting a Test:

Currently there are two input test sequences: a step input and a random input. To perform a transient response on either the azimuth or the elevation servos, simply type into the host terminal connected to the anglescan Z80 computer: **S a=xxxx** for the azimuth servo or **S e=xxxx** for the elevation servo, where xxxx is the step size in hundredths of degrees (refer to section 2.10 on the user menu interface). Alternatively a PRBS response can be performed on either the azimuth or the elevation servo by typing in the command: **P a** for the azimuth servo, or **P e** for the elevation servo (refer to the user menu interface in section 7.7A).

The random input test is used primarily for identification purposes, whereas the transient test is used for model verification purposes. At the completion of the test, the program dumps the measurements into an IBM PC which stores the data into a file for later processing by the system identification programs. The data file is essentially a text file consisting of two colums of data representing u_k and y_k sequences respectively.

Running the Identification programs:

The program **SYSTIDE.C** was developed for system identification, a copy of the C source code is included in the appendix 7.4B. The program implements the following identification schemes:

(1) Recursive least squares with optional forgetting factor.
(2) Non- Recursive least squares.
(3) Extended recursive least squares with optional forgetting factor.
(4) Higher order models.(more unknown coefficients)
(6) Reduced order models.(less unknown coefficients)

Additionally the program is able to determine the presence of input offsets and disturbances. The program has been tested and verified with simulated test data. A description of each identification scheme and experimental results is provided in the sections below.

This identification program has no graphical output. However there are 2 other programs which carry out the identification and also produce graphical outputs to both the screen (VGA type) and the HP plotter for documenting. The programs are **SYSTIDE1.C** and **SYSTIDE2.C**, the results from the graphical output (HP plotter) is provided in the thesis. The last two programs have not being included in the appendix but implement identical system identification algorithms.

In the next few sections, a description of each model structure and identification algorithm is given along with the experimental results obtained.

[3.5] Ordinary Least Squares (LSE):

The simplest form of identification scheme implemented operates on a large set of measurements simultaneously to obtain the required estimates of model coefficients. It assigns equal weight to each measurement with no forgetting factor.

Description: Given the linear equation of the model to be identified as having the general form:

$$\frac{Y(z)}{U(z)} = \frac{b_1 z + b_2}{z^2 - a_1 z - a_2} \tag{3.29}$$

From equation 3.29, the recursive function in discrete time then becomes:

$$y_k = a_1.y_{k-1} + a_2.y_{k-2} + b_1.u_{k-1} + b_2.u_{k-2} + e_k \tag{3.30}$$

In matrix form, the system of linear equations for N measurements then becomes:

$$\begin{bmatrix} y_2 \\ y_3 \\ \cdot \\ \cdot \\ \cdot \\ y_N \end{bmatrix} = \begin{bmatrix} y_1 & y_0 & u_1 & u_0 \\ y_2 & y_1 & u_2 & u_1 \\ & & \cdot & \\ & & \cdot & \\ & & \cdot & \\ y_{N-1} & y_{N-2} & u_{N-2} & u_{N-1} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ b_1 \\ b_2 \end{bmatrix} + \begin{bmatrix} e_2 \\ e_3 \\ \cdot \\ \cdot \\ \cdot \\ e_N \end{bmatrix} \tag{3.31}$$

Re-write in matrix notation:

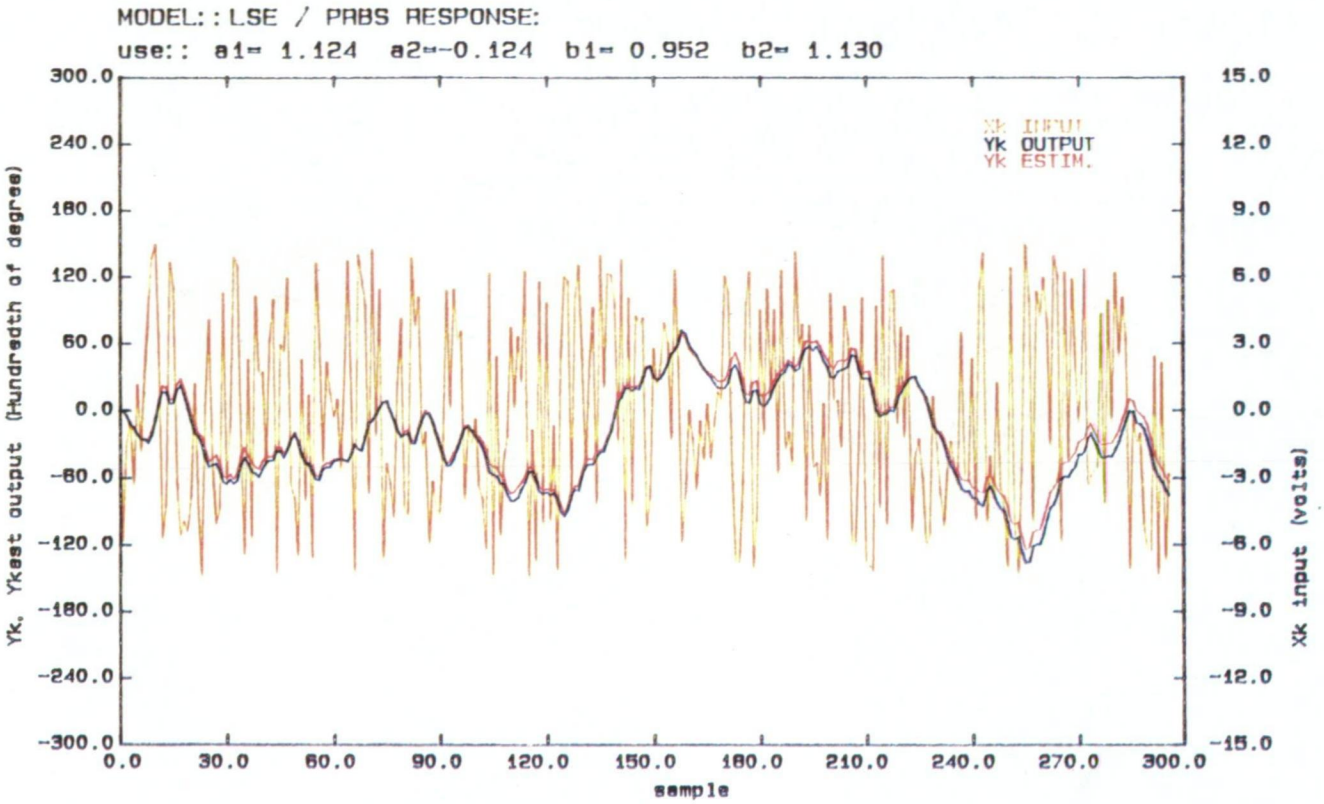
$$\underline{Y} = \mathbf{H}\,\underline{\theta} + \underline{e} \tag{3.32}$$

To minimize the sum of the errors squared (see eqn 3.10), the solution is given by the matrix product: (refer to appendix 7.3A):

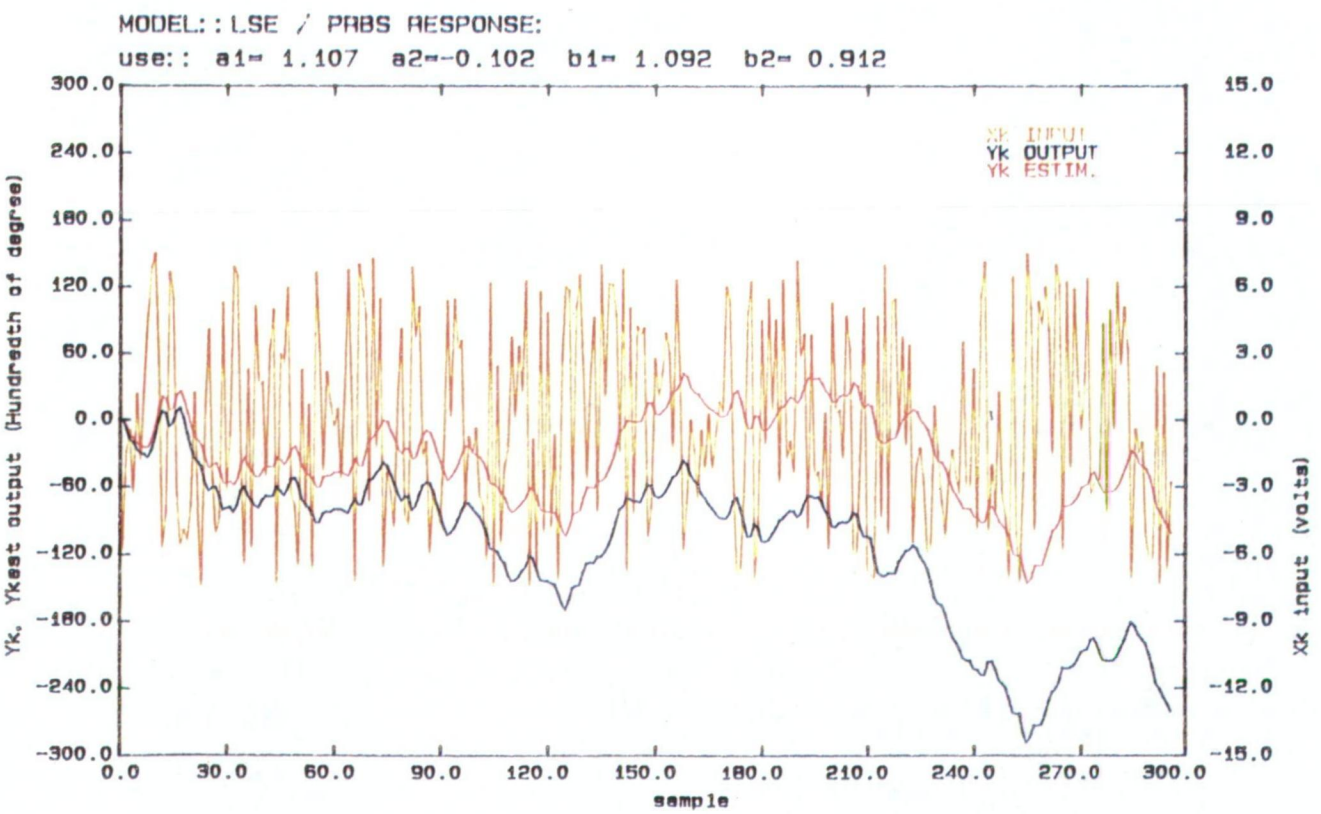
$$\hat{\underline{\theta}} = \left[\mathbf{H}^T \mathbf{H}\right]^{-1} \mathbf{H}^T.\underline{Y} \tag{3.33}$$

Experimental Results: The program "SYSTIDE.C" uses a subroutine called: **LSE()** to evaluate the above expression. We ran the program on measurements obtained from the anglescan system. Graphs 3.1A and 3.1B on the following page illustrate the results for both azimuth and elevation servosystems and the coefficients evaluated by the program. No input offset is assumed in both instances. From the elevation plot, it is evident that an external disturbance (torque) is present due to gravity. Offset disturbance is considered in section 3.8. Experimental results are tabulated in Tables 3.15A and B.

Graph.3.1A Identification LSE: [Azimuth] a1=1.12445 a2=-0.12369 b1=0.95166 b2=1.13000



Graph.3.1B Identification LSE: [Elevation] a1=1.10652 a2=-0.10235 b1=1.09214 b2=0.91235



[3.6] Recursive Least Squares (RLSE):

Recursive least squares solves for the unknown model coefficients recursively rather than by a single matrix evaluation as previously with equation 3.32.
For each set of new measurements obtained, the unknown model coefficients are evaluated. Ideally, the coefficients should progressively approach some steady-state value. This method has the added advantage of providing new estimates for each new set of measurements obtained thus able to track parameter variations with time.

Parameter tracking is achieved by the inclusion of a weighting factor which increases with time (exponential forgetting factor) placing less importance on earlier measurements and progressively adapting to current measurements.

One disadvantage however with this method is due to the convergence of the covariance matrix which causes measurements obtained at a later time to carry less weight (refer to section 3.11).

Description: Consider the following equation:

$$\frac{Y(z)}{U(z)} = \frac{b_1 z + b_2}{z^2 - a_1 z - a_2} \tag{3.33}$$

recursively:

$$y_k = a_1.y_{k-1} + a_2.y_{k-2} + b_1.u_{k-1} + b_2.u_{k-2} + e_k \tag{3.34}$$

given a new set of measurements y_{k+1}, then a better estimate from the previous estimate is:

$$\begin{aligned} \underline{K}_{k+1} &= \mathbf{P}_k \underline{H}_{k+1}^T (\delta + \underline{H}_{k+1} \mathbf{P}_k \underline{H}_{k+1}^T)^{-1} \\ \underline{\hat{\theta}}_{k+1} &= \underline{\hat{\theta}}_k + \underline{K}_{k+1} (y_{k+1} - \underline{H}_{k+1} \underline{\hat{\theta}}_k) \\ \mathbf{P}_{k+1} &= (\mathbf{P}_k - \underline{K}_{k+1} \underline{H}_{k+1} \mathbf{P}_k)/\delta \end{aligned} \tag{3.35}$$

For this model structure, the terms in equations 3.35 are defined by the following matrices:

where: $\hat{\underline{\theta}}_{k+1} = [a_1 \; a_2 \; b_1 \; b_2]^T$

$\underline{H}_{k+1} = [y_k \; y_{k-1} \; u_k \; u_{k-1}]$

$\mathbf{P}_{k+1}$ = [4x4] matrix

$\underline{K}_{k+1}$ = [4x1] matrix

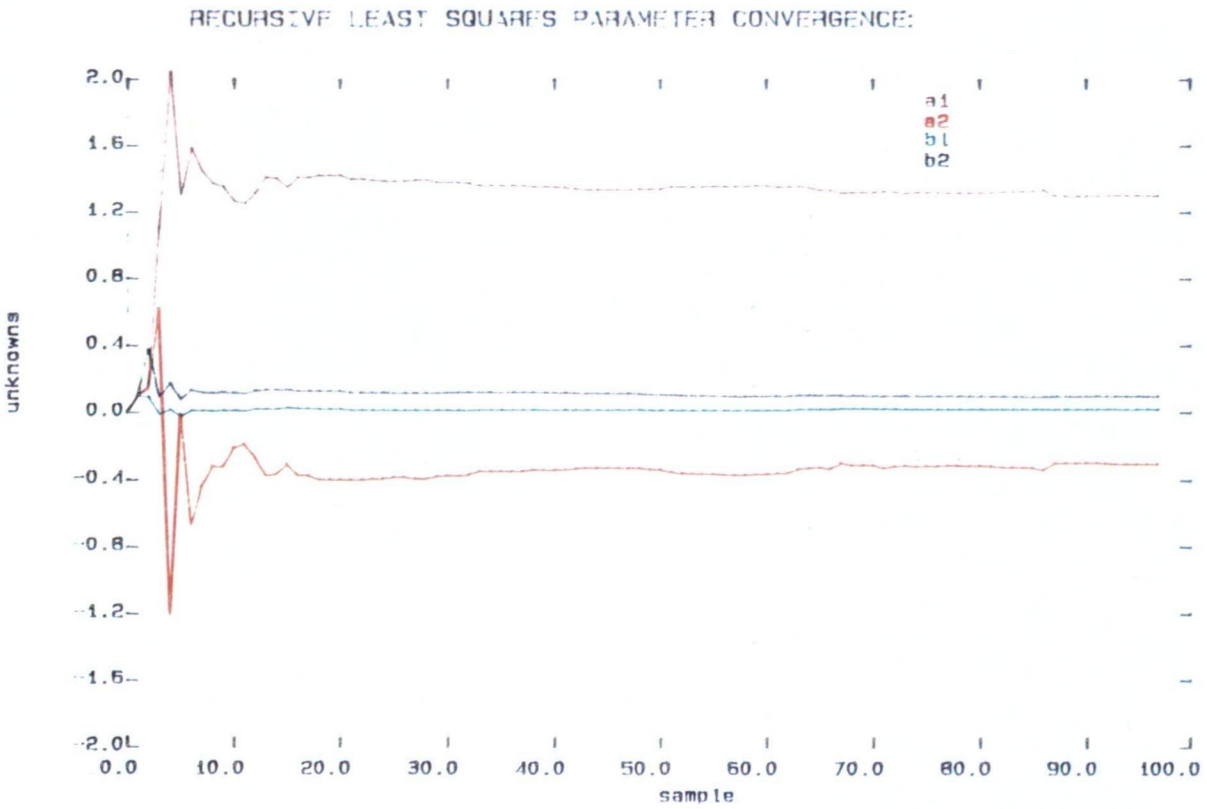
The equations are evaluated in the above order. Note that the recursive algorithms we initially defined $\mathbf{P}_k$ to be a diagonal matrix, and $\underline{\theta}_k$ to be the initial value of the estimates. The symbol δ is the forgetting factor, set between 0.95 to 0.99, must be less than or equal to 1.0.

Experimental Results: The program **"SYSTCAL.C"** contains the subroutine: **RLSE()** which implements the above identification scheme. Results from this method are tabulated in Table-3.15 and a graphical plot of the measured response compared to the predicted response with the unknown values of a_1, a_2, b_1, b_2 as given by Table-1. No input offset was calculated for the elevation model as in section (3.5) above.

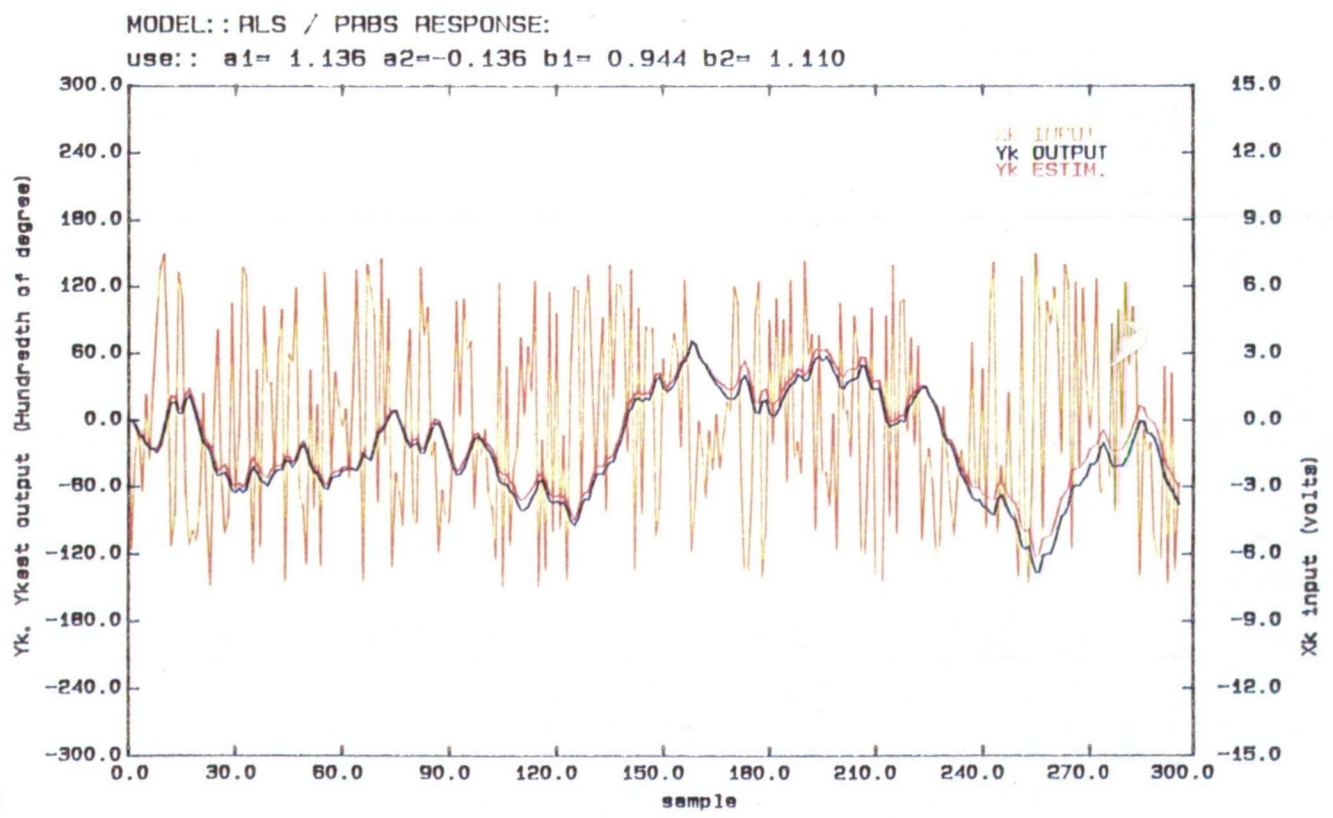
Note that the coefficients and the error compared with the LSE method is approximately the same.

Graph 3.2A below shows a typical parameter convergence plot of the four unkowns: {a_1, a_2, b_1, b_2}. Graph 3.2B and 3.2C show the actual measured response y_k and the estimated response from the model.

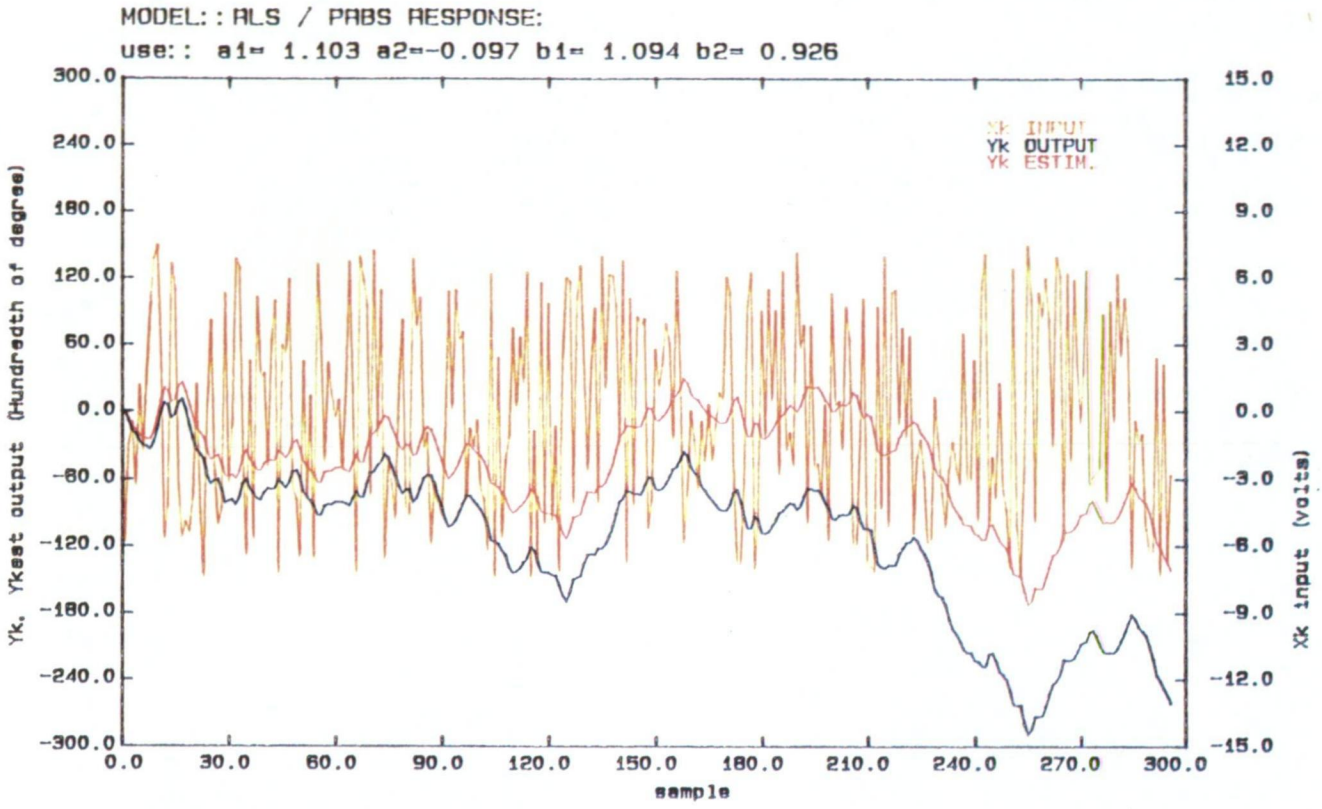
Graph 3.2A Identification: RLSE: Recursive Least Squares Parameter Convergence:



Graph.3.2B Identification RLSE [Azimuth] a1=1.13648 a2=-0.13620 b1=0.94357 b2=1.11002



Graph.3.2C Identification RLSE: [Elevation] a1=1.01247 a2=-0.09683 b1=1.09382 b2=0.92603



[3.7] Extended Recursive Least Squares:

Extended least squares incorporates a noise structure modelled by a linear time invariant filter (first order) with zero mean gaussian input. This type of identification provides better estimation in the presence of measurement noise.

Description: Consider the following system model with the added noise structure:

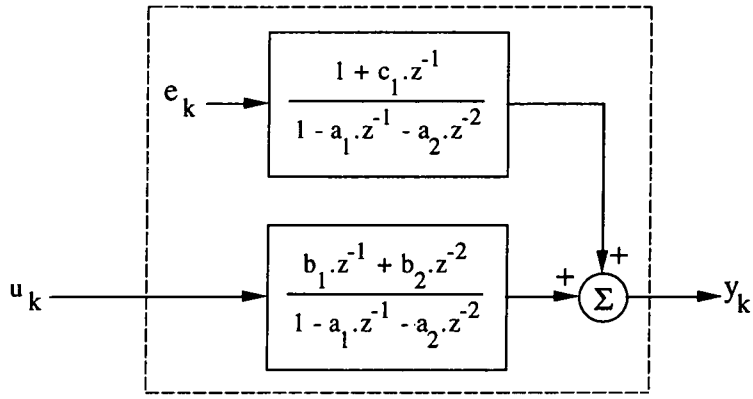


Fig.3.11 Extended Least Squares Model

There are 5 unkowns to solve: $\{a_1, a_2, b_1, b_2, c_1\}$. The recursive equation becomes:

$$y_k = a_1.y_{k-1} + a_2.y_{k-2} + b_1.u_{k-1} + b_2.u_{k-2} + e_k + c_1.e_{k-1} \qquad (3.36)$$

The recursive equations are shown in 3.36B below, with the variables containing an additional noise term c_1, thus:

$$\underline{K}_{k+1} = \mathbf{P}_k \underline{H}_{k+1}^T (\delta + \underline{H}_{k+1} \mathbf{P}_k \underline{H}_{k+1}^T)^{-1}$$

$$\underline{\hat{\theta}}_{k+1} = \underline{\hat{\theta}}_k + \underline{K}_{k+1}(y_{k+1} - \underline{H}_{k+1}\underline{\hat{\theta}}_k)$$

$$\mathbf{P}_{k+1} = (\mathbf{P}_k - \underline{K}_{k+1}\underline{H}_{k+1}\mathbf{P}_k)/\delta \qquad (3.36B)$$

where: $\underline{\hat{\theta}}_{k+1} = [a_1 \; a_2 \; b_1 \; b_2 \; c_1]^T$

$\underline{H}_{k+1} = [y_k \; y_{k-1} \; u_k \; u_{k-1} \; e_{k-1}]$

$\mathbf{P}_{k+1}$ = [5x5] matrix

$\underline{K}_{k+1}$ = [5x1] matrix

The algorithm proceeds as described in section [3.6], with the initial value of $e_k=0$ at k=0. At completing each iteration we calculate the error term by subtracting the observed value from the expected value of y_k:

$$e_k = y_k - H_k.\theta_k \tag{3.37}$$

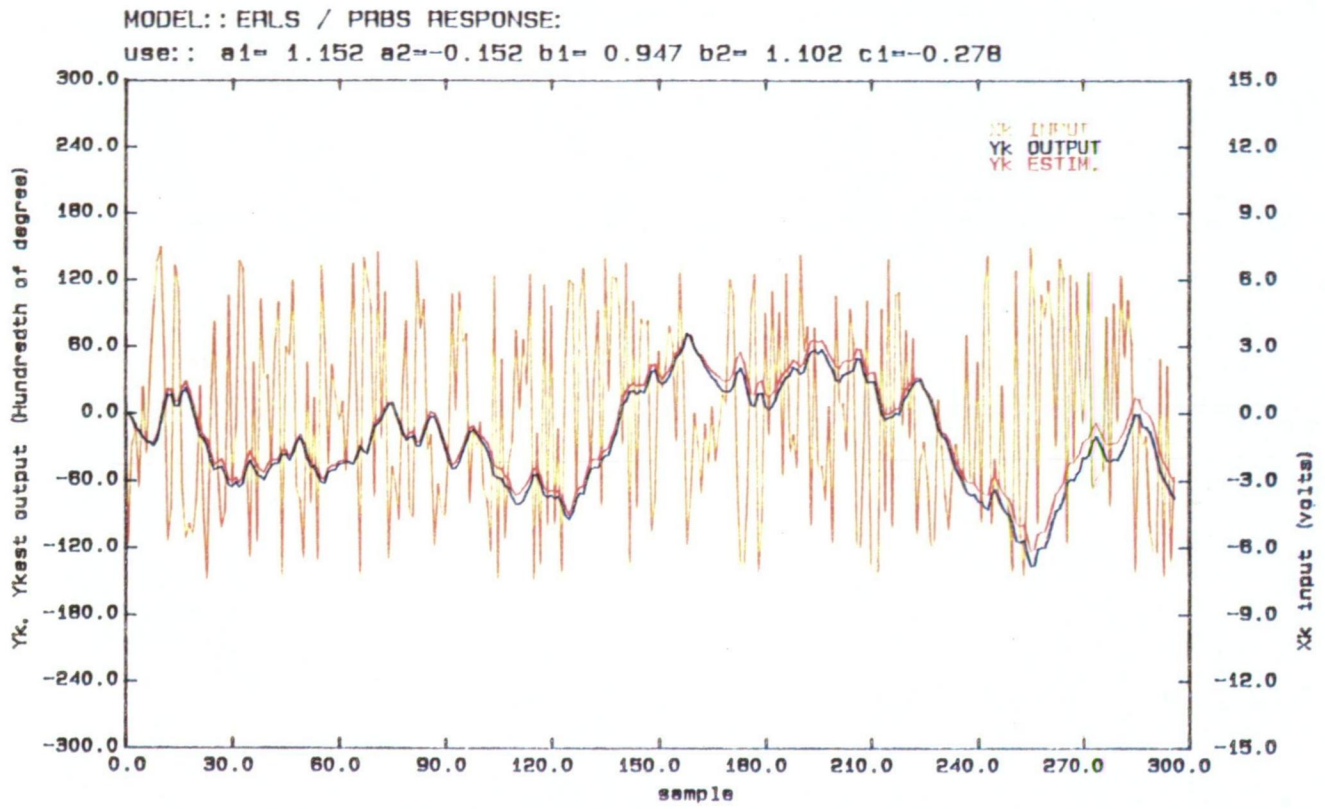
The value of e_k is subsequently inserted on the next iteration as the previous value of e_{k-1}.

Experimental Results: The program "**SYSTCAL.C**" contains the subroutine: **ERLSE()** to determine the five unkown estimates, results are tabulated in Table-3.15A,B.

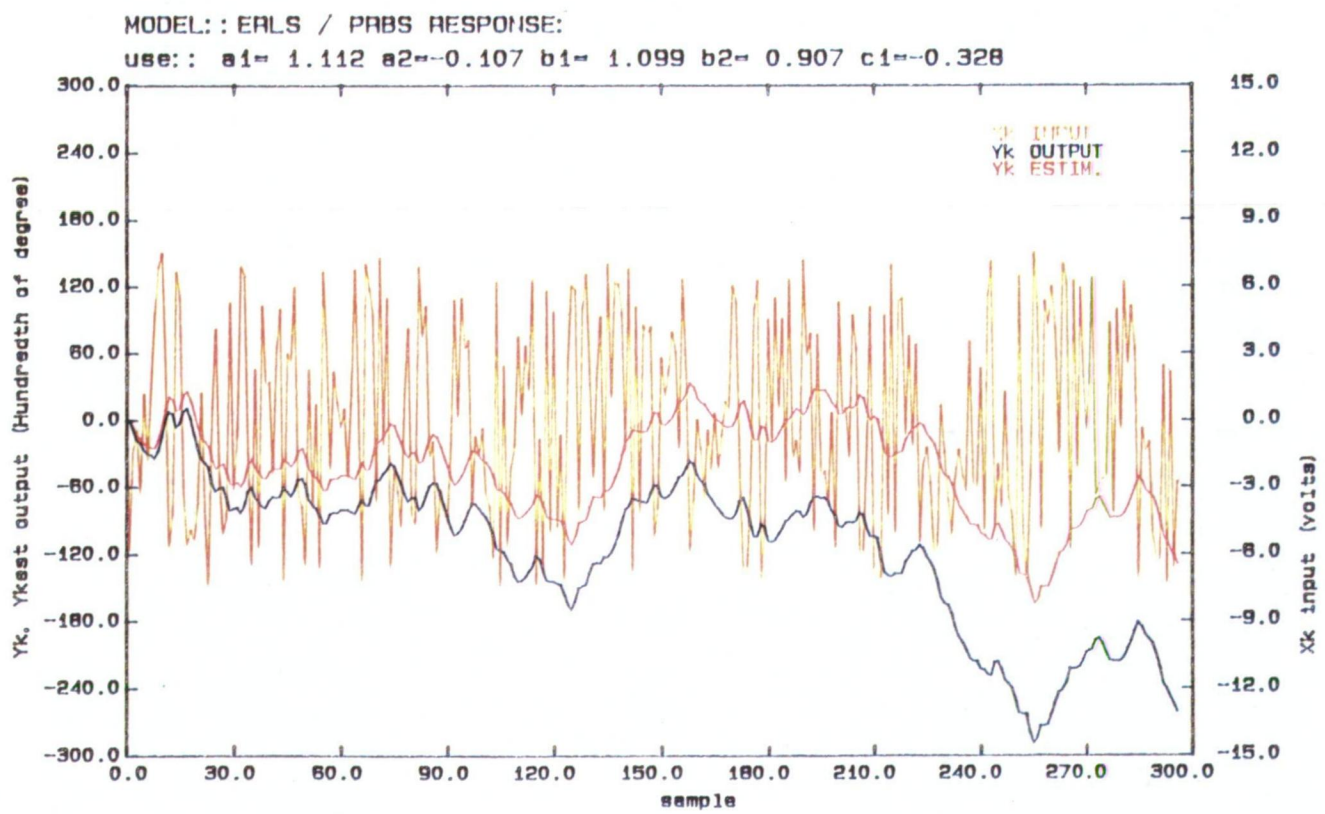
On the following page, we plot the calculated output of y_k and the measured output of y_k for a PRBS input source.
Note that there is little improvement over the previous method of identification with recursive least squares indicating that noise is not a significant contributing factor.

Note that on all of the identification methods used so far, the azimuth model accurately follows the actual measurements. However the elevation servo exhibits a DC component superimposed on the output. This DC disturbance is due to torque produced by the gravity term which is in effect an input offset to the model.

Graph.3.3A Identification ERLSE: [Azimuth] a1=1.1520 a2=-0.1519 b1=0.9464 b2=1.1021 c1=-0.2780



Graph.3.3B Identification ERLSE: [Elevation] a1=1.1118 a2=-0.1067 b1=1.1.0993 b2=0.9073 c1=-0.3280



[3.8] Least Squares with Offset:

The presence of both input and output offsets or slow DC disturbances can be included into the model to be identified. Conceptually, the output DC disturbance could be caused by gravity, which is the case for the elevation servosystem.
The DC offset disturbances can be found by least squares in a similar way by simply including a constant bias term in the model. Consider the following system to be identified where d_1 and d_2 define the input and output offset respectively:

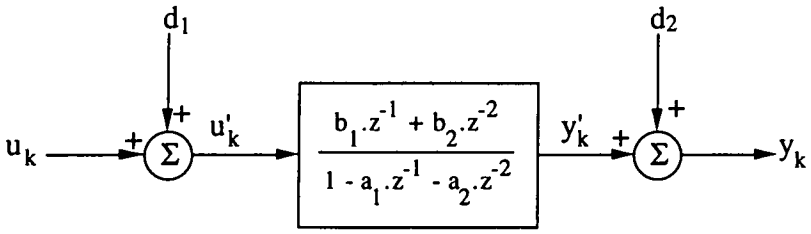


Fig.3.12 Model with Input and Output Offsets

The transfer function for the above model becomes:

$$y'_k = a_1.y'_{k-1} + a_2.y'_{k-2} + b_1.u'_{k-1} + b_2.u'_{k-2} \qquad (3.38)$$

Noting the terms $y'_k=y_k-d_2$ and $u'_k=u_k+d_1$, substituting into equation 3.38 above we get the following equation:

$$y_k = a_1.y_{k-1} + a_2.y_{k-2} + b_1.u_{k-1} + b_2.u_{k-2} + d_1(b_1 + b_2) - d_2(a_1 + a_2 - 1) \qquad (3.39)$$

The last two terms combine into one to form an overall offset component. Note that the denominator of the above transfer function contains a pole at z=1, this is evident from the previous identification results and from the servomotor model developed in section [3.3] (refer to equation 3.27). If this is the case, then the term $(a_1 + a_2 -1) = 0$ vanishes. In general, if the system is a type 1 system (pole at s=0 or z=1) then it would not be possible to determine the output offset using the above method.

Thus the recursive equation becomes:

$$y_k = a_1.y_{k-1} + a_2.y_{k-2} + b_1.u_{k-1} + b_2.u_{k-2} + \hat{d}_1 \qquad (3.40)$$

where the terms in the recursive least squares equation are given by:

$$\text{where:} \quad \hat{\underline{\theta}}_{k+1} = [a_1 \; a_2 \; b_1 \; b_2 \; \hat{d}_1]^T$$

$$\underline{H}_{k+1} = [y_k \; y_{k-1} \; u_k \; u_{k-1} \; 1]$$

$$\hat{d}_1 = d_1(b_1 + b_2)$$

note the addition of the offset term estimate d_1 which is included as an unknown parameter in parameter vector $\underline{\theta}_{k+1}$ and regression vector $\underline{H}_{k+1}$.

Experimental Results: The program "**SYSTCAL.C**" contains the subroutine: **RLSEO()** to implement the above identification scheme. Results are given in Table-3.15A,B. Note the identification produces much more accurate results for the elevation servosystem.

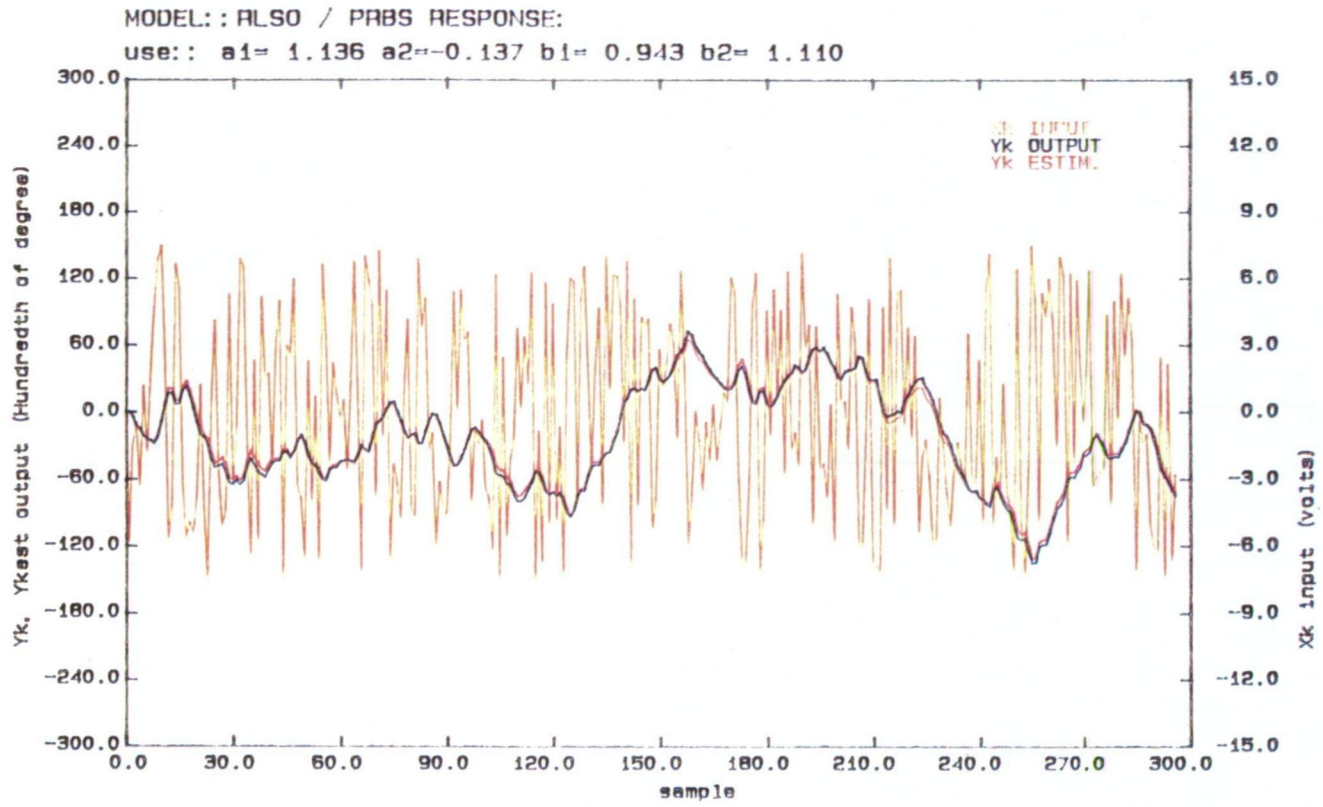
Graphs 3.4A and 3.4B on the following page compare results from the simulation model and the measurements. The elevation model in this instance follows the measurement data better than the previous estimates.

Extended Recursive Least Squares with Offset:

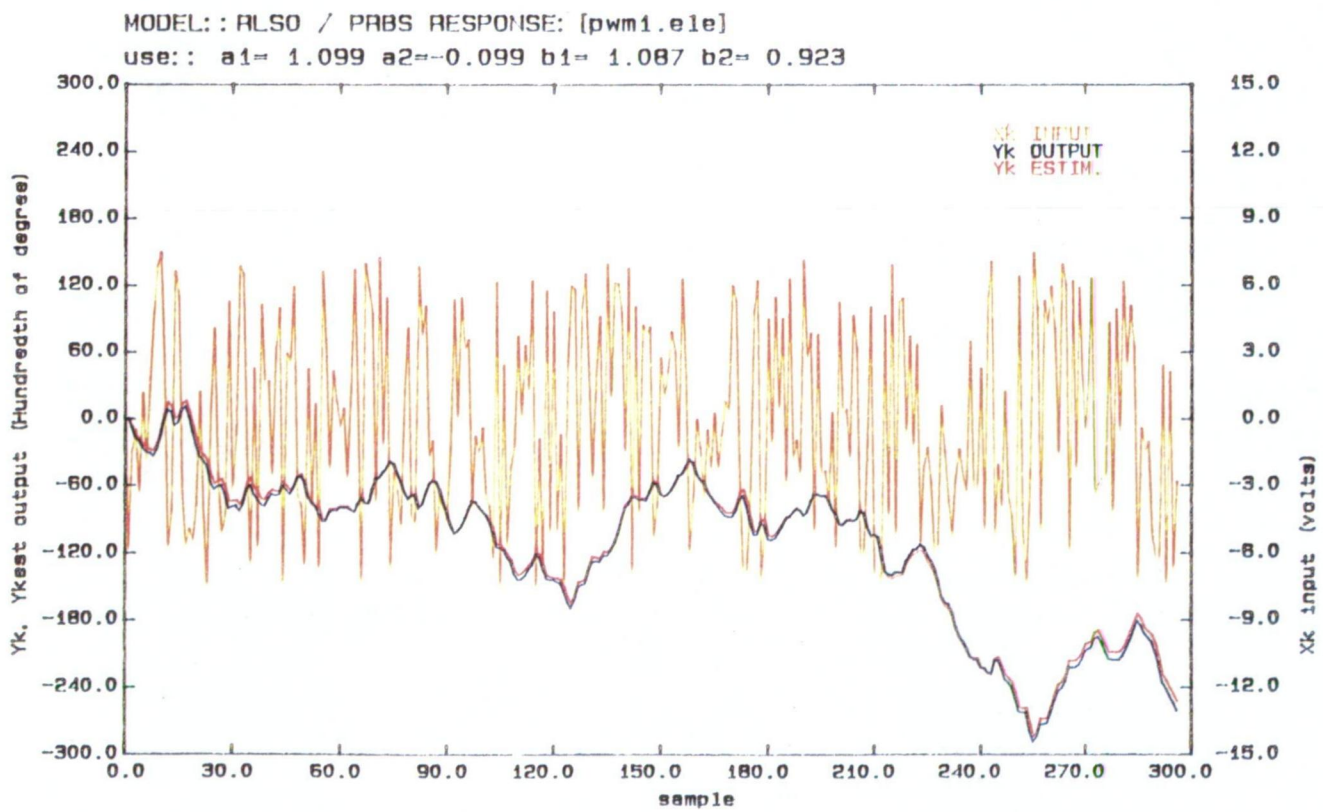
The same procedure was applied using Extended Least Squares solution with input offset, the subroutine **ERLSO()** performs the calculations, results are given in Table 3.15A,B.

The error term in table 3.15A and 3.15B shows that with the inclusion of an offset term, the elevation error is reduced by a factor of 20 times over the previous identification models, similarily there is also a marginal improvement in the azimuth servo system model.

Graph.3.4A Identification RLSEO: [Azimuth] a1=1.1364 a2=-0.1365 b1=0.9431 b2=1.1096 d1=-0.0403



Graph.3.4B Identification RLSO: [Elevation] a1=1.0989 a2=-0.0987 b1=1.0870 b2=0.9203 d1=-0.5769



[3.9] Higher Order Model:

The 4 parameter model (discussed in section 3.7 above) can be easily modified to accommodate 6 parameters. The six parameter model is more complex but may not necessarily be more accurate. We investigate this model illustrated by Fig.3.13 below:

Theory: The model structure illustrated below includes one extra pole and zero:

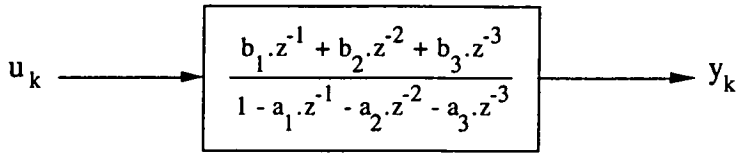


Fig.3.13 Six Parameter Model

Gives the recursive time domain equation:

$$y_k = a_1.y_{k-1} + a_2.y_{k-2} + a_3.y_{k-3} + b_1.u_{k-1} + b_2.u_{k-2} + b_3.u_{k-3} \qquad (3.41)$$

where the unknown vector and covariance vector are defined as:

$$\underline{\hat{\theta}}_{k+1} = [a_1 \; a_2 \; a_3 \; b_1 \; b_2 \; b_3]^T \qquad (3.42A)$$

$$\underline{H}_{k+1} = [y_k \; y_{k-1} \; y_{k-2} \; u_k \; u_{k-1} \; u_{k-2}] \qquad (3.42B)$$

Apply the recursive least squares estimation defined by equations 3.35 to solve for the six unknowns.

Experimental Results:

The program "**SYSTCAL.C**" contains the subroutine: **RLSE6()** to solve the above identification scheme. Results are tabulated in Table3.15A,B. Note that this method does not offer much improvement over the existing four parameter model at the expense of added complexity. This indicates that the four parameter model is sufficiently accurate as the mean squared error is the same for the two models. Note that graphical plots are not available for this particular model and only the results are tabulated.
The addition of another pole and zero is the same as including the electrical time constant of the L/R combination of the armature winding which was originally ignored in the development of the servo-motor model in section 3.3.

[3.10] Reduced Coefficients Model:

Rather than increasing the number of unknown coefficients of the model, the complexity of the model can be reduced to fewer unknowns.

Theory: From servomotor models developed in section [3.3], and referring to equation 3.27, the model has a pole fixed at z=1. This reduces the number of unknowns to three only. Consider the transfer function with the structure:

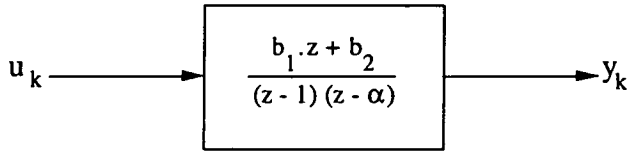


Fig.3.14 Reduced Order Model

Where the unkowns are: $\{b_1, b_2, a\}$. Expanding the denominator polynomial and dividing by through by: z^2

$$H(z) = \frac{b_1.z^{-1} + b_2.z^{-2}}{1 - (\alpha+1).z^{-1} + \alpha.z^{-2}} \tag{3.43}$$

Recursively, the equation becomes:

$$y_k = \alpha.(y_{k-1} - y_{k-2}) + y_{k-1} + b_1.u_{k-1} + b_2.u_{k-2} \tag{3.44}$$

Thus the unkown vector and covariance vector are now defined as:

$$\underline{\hat{\theta}}_{k+1} = [\alpha \;\; 1 \;\; b_1 \;\; b_2]^T \tag{3.45A}$$

$$\underline{H}_{k+1} = [y_k - y_{k-1} \;\; y_k \;\; u_k \;\; u_{k-1}] \tag{3.45B}$$

Experimental Results:

The program **"SYSTIDE.C"** contains the subroutine: **RLSE3()** to perform the above identification scheme. The results are tabulated in Table 3.15A,B below. From the results, the method does not offer improvement over the previous four parameter model described in section 3.5. Refering to Table 3.15, all other identification algorithms locate a pole close to z=1 thus verify that the system is type-1. Graphical results are not available for this model.

[3.11] Constant Trace and Forgetting Factor:

Two important aspects of the recursive least squares identification technique which provide similar effects upon the convergence of the algorithm are (a) Constant trace, (b) Forgetting factor.

(a) Constant Trace: Otherwise known as Covariance Resetting, it is thoroughly described in a paper by M.T.Tham, 1988 (see reference 3.11). The ability to effectively track time varying process parameters by on-line recursive parameter identification algorithms is lost once the convariance matrix has converged. Furthermore, the rate at which parameter tracking occours depends upon this matrix.

The covariance matrix thus effectively represents a gain vector, the elements of the gain vector determine the direction (via their sign) and rate of parameter tracking (via their magnitude). Once the unknown parameters converge, the elements of the covariance matrix decay to a small value preventing the ability of the recursive algorithm to further track parameter variations. The constant trace operates on the main diagonal elements of the covariance matrix:

$$\mathbf{H} = \begin{bmatrix} \sigma_{11} & \cdots & \cdots \\ \cdots & \sigma_{22} & \cdots \\ \vdots & & \\ \cdots & \cdots & \sigma_{NN} \end{bmatrix} \tag{3.46}$$

A number of methods exist for modifying this matrix, either re-initialization of all diagonal elements or modification. Additionally, we can either modify elements individually, ie those below a certain threshold, or operate on the entire diagonal. For the later, the user can input a value α=trace value, then if the trace of the matrix is below this value, the diagonal is completely rescaled.
The common aim of any of the covariance resetting techniques is to increase the magnitude of the diagonal elements whilst ensuring that the parameter change is 'smooth'. Define the trace of the matrix **H** to be tr[**H**]:

$$tr[\mathbf{H}] = \sum_{j=1}^{N} \sigma_{jj} \tag{3.47}$$

if the trace of H falls below the user specified value of α, the diagonal elements of the matrix are rescaled, thus:

$$\sigma_{jj} = \sigma_{jj} \frac{\alpha}{tr[H]} \tag{3.48}$$

Other methods exist such as resetting all diagonals to α, or individually if any diagonal element is below the user specified. Each diagonal element controls the 'trackability' of its corresponding parameter, and is an indication of the estimator performance.

We have briefly mentioned covariance resetting as a method which avoids possible incorrect convergence. Fortunately this problem has not occoured with system identification in our case.

(b) Forgetting Factor: Has been used in all identification schemes implemented above, but with little improvement upon the convergence and results.
The emphasis is that more recent data is weighted considerably more, whilst old data is considered obsolete. It operates by applying an exponential decaying factor to each measurement, thus each measurement is weighted:

$$\beta(t,k) = \lambda^{t-k} \tag{3.49}$$

where the forgetting factor: $0< \lambda < 1$, up to measurement at time t. This method has the disadvantages of producing estimator 'blowups' or 'bursting' during periods of low system excitation however.
Different values of forgetting factor between 0.95 to 1.0 were used during recursive identification and found to have no effect upon the values of the model parameters.

Table 3.15: Identification results for both azimuth and elevation servomotor models using the system identification program: SYSTIDE.C for all the model structures described:

TABLE 3.15A Elevation Servo Motor

MODEL	a1	a2	a3	b1	b2	b3	c1	d1	ERROR	POLE 1	POLE 2	ZERO
RLSE	1.01247	-0.09683	---	1.09382	0.92603	---	---	---	67.71	1.00624	0.09623	-0.84660
ERLSE	1.11188	-0.10671	---	1.09938	0.90737	---	-0.3280	---	71.85	1.00579	0.10610	-0.82535
LSE	1.10652	-0.10235	---	1.09214	0.91235	---	---	---	81.21	1.00464	0.10188	-0.83538
RLSE6	0.79079	0.29343	-0.07717	1.10300	1.26395	0.25134	---	---	66.81	x.xxxxx	x.xxxxx	-x.xxxxx
RLSE3	1.09684	-0.09684	---	1.09382	0.92603	---	---	---	100.35	1.00000	0.09684	-0.84660
RLSEO	1.09899	-0.09875	---	1.08705	0.92305	---	---	-0.5769	3.11	1.00027	0.09873	-0.84913
ERLSEO	1.11044	-0.10970	---	1.08926	0.90639	---	-0.3550	-0.5099	6.16	1.00083	0.10961	-0.83212
LSEO	1.10369	-0.10288	---	1.08858	0.91219	---	---	-0.5256	3.56	1.00090	0.10279	-0.83796

TABLE 3.15B. Azimuth Servo Motor

MODEL	a1	a2	a3	b1	b2	b3	c1	d1	ERROR	POLE 1	POLE 2	ZERO
RLSE	1.13648	-0.13620	---	0.94357	1.11002	---	---	---	-6.70	1.00032	0.13615	-1.17640
ERLSE	1.15202	-0.15190	---	0.94649	1.10210	---	-0.2780	---	-6.92	1.00014	0.15188	-1.16441
LSE	1.12445	-0.12369	---	0.95166	1.13000	---	---	---	-5.19	1.00088	0.12358	-1.18739
RLSE6	0.85445	0.23454	-0.08942	0.95337	1.37134	0.27471	---	---	-7.93	0.99959	0.23520	-1.19786
RLSE3	1.13620	-0.13620	---	0.94357	1.11001	---	---	---	-7.49	1.00000	0.13620	-1.17640
RLSEO	1.13647	-0.13659	---	0.94312	1.10960	---	---	-0.0403	-1.03	0.99986	0.13661	-1.17652
ERLSEO	1.15159	-0.15164	---	0.94594	1.10255	---	-0.2700	-0.0266	-2.79	0.99994	0.15165	-1.16557
LSEO	1.12434	-0.12391	---	0.95113	1.12959	---	---	-0.0350	-0.04	1.00050	0.12384	-1.18763

Notes: Fig.3.15: Summary of Results:

The program **SYSTIDE.C** is listed in the appendix 7.4B. The program implements all of the identification schemes described below, each model is implemented by a separate c function, for example the RLSE model is implemented by the function RLSE(), etc...

(a) Model acronyms: The following model acronyms are refered by table 3.15 and the number of unknown parameters in each model:

Name:	Description:	Unknowns
RLSE:	Recursive Least Squares Estimation	4
ERLSE:	Extended Recursive Least Squares Estimation	4
LSE:	Least Squares Estimation (non recursive)	4
RLSE6:	Recursive Least Squares Estimation 6 parameters	6
RLSE3:	Recursive Least Squares Estimation 3 parameters	3
RLSEO:	Recursive Least Squares Estimation+Offset	5
ERLSEO:	Extended Recursive Least Squares Estimation+Offset	6
LSEO:	Least Squares Estimation (non recursive)+Offset	5

(b) Model Coefficients: The $\{a_1, a_2, a_3\}$ define the coefficients of the denominator polynomial. The only identification scheme to use all three coefficients is the higher order model described in section 3.8, the remaining models use only two coefficients, in which case the entries are unused. Similarily for the coefficients $\{b_1, b_2, b_3\}$ of the numerator polynomial.
The coefficient $\{c_1\}$ defines the linear noise filter structure which is only solved by the Extended least squares algorithm (ERLSE, ERLSEO), all remaining entries are unused.

The coefficient $\{d_1\}$ defines the equivalent input offset to the system model which is solved only by the last three algorithms (RLSEO, ERLSEO, LSEO), all other entries remain unused. Note this value is large for the elevation servo compared with the azimuth servo. To convert the value of d_1 into 'equivalent' volts offset on the input to the servo system, the following relationship can be used:

$$Vos = \frac{d_1'}{(b_1 + b_2)} Vcc \qquad (3.50)$$

where Vcc is the PWM pulse amplitude in volts, typically 18V, b_1 and b_2 are the model coefficients.

(c) Error term: The error is computed as the difference (squared) between the measured output and the estimated output from the particular model and its corresponding coefficients. Thus for N samples:

$$\text{error} = \frac{1}{N}\sum_{k=1}^{N}\left[y_k - \hat{y}_k\right]^2 \qquad (3.51)$$

(d) Poles/Zeros: All identification schemes calculate the location of the poles and zeros. Note that in all identification schemes one pole is always located at z=1.0.
The simpler 4 parameter model is assumed for simulation and control system design purposes, with input offset on the elevation system only.

The two systems are not identical due to differences in load dynamics ie: inertia and friction, but the servomotors and drive electronics is identical in both instances.
Figures 3.18A and 3.18B show the poles and zeros chosen for each model which give a reasonably small error squared (equation.3.51).

[3.12] Identification of Coulomb Friction/Stiction:

So far, system identification has not included the non linear effects of stiction and coulomb friction. A paper by R.Stanway 1986 (ref.3.9) shows how to incorporate the combined effects coulomb and viscous friction in the least squares identification technique:

(a) Stiction: Stiction is the frictional force which prevents motion until the driving force exceeds some minimum value.

(b) Coulomb Friction: Coulomb friction is a constant frictional drag which opposes motion with a magnitude independent of velocity.

(c) Actuator Saturation: Actuator saturation occours at 100% duty cycle on the PWM drives and is included in control system simulation.

Fig.3.16 illustrates the combined effects of stiction, coulomb friction and actuator saturation. The input and output can refer to either voltage applied or torque since the two are interchangable ie: T=Va/Ra.Kt, where Va=armature voltage, Ra=armature resistance, Kt=torqure coefficient. The effects of the three model non-linearities can easily be included during computer simulation .

Stiction and coulomb friction (combined) can be identified with the system initially at rest. By gradually increasing the applied duty cycle until motion commences, at this point the input is just sufficient to overcome stiction Vs. This is tested in both forward and reverse modes.

Coulomb friction is determined with the system initially in motion. By gradually reducing the applied duty cycle until motion ceases, at this point the input is equal to the coulomb friction.

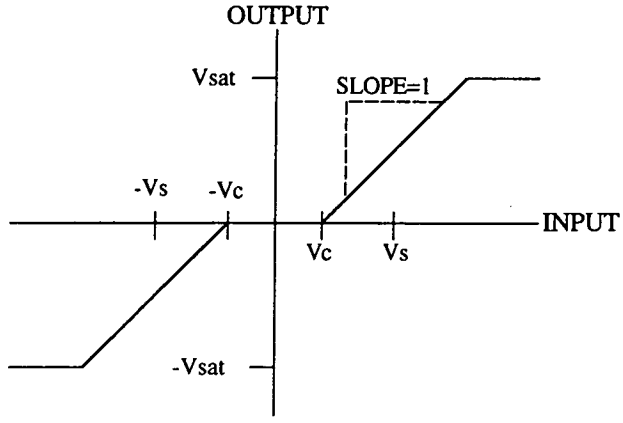


Fig.3.16 Combined Effects of Stiction, Coulomb and Saturation

Measurements indicate the presence of stiction is negligible, the table below illustrates the cutoff points at which stiction and coulomb friction occour.

STICTION...: Vs=0.30V
COULOMB...: Vc=0.25V

The model is simplified by ignoring stiction and only assume the presence of coulomb friction. This is effectively deadband as shown in Fig.3.16.

Fig.3.17 below illustrates the equivalent model used for both azimuth and elevation servosystems, the elevation model includes an input offset term added.

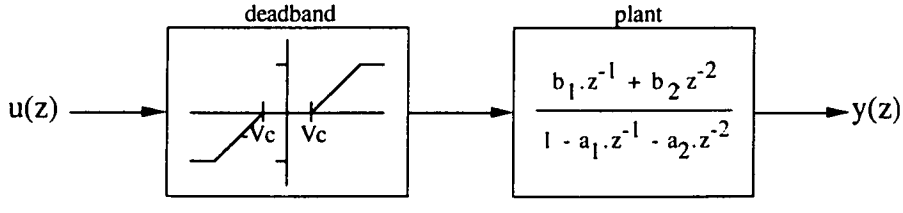


Fig.3.17 Model with Deadband and Saturation

[3.13] Summary of Azimuth and Elevation Models:

The following plant models are adopted for control system design in the z plane. Table-3.15 originaly defined u_k in units of volts instead of binary bn.
Note that the numerator coefficients b_1 and b_2 have been rescaled from volts to binary bn.

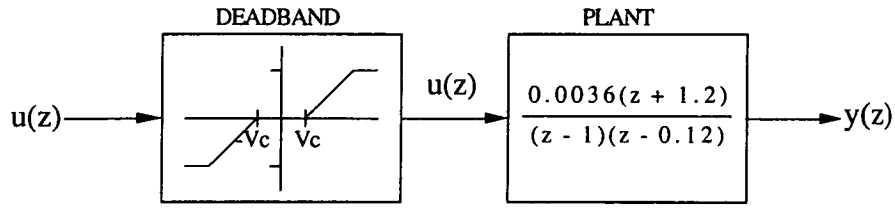


Fig.3.18a Azimuth servosystem model

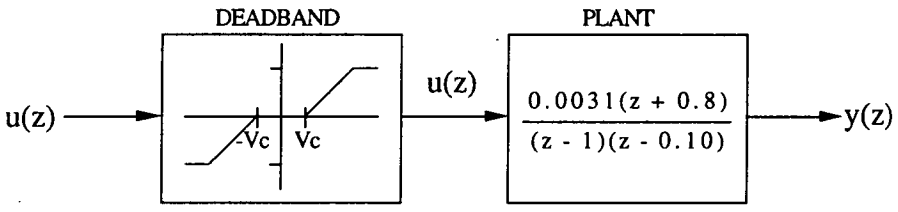


Fig.3.18b Elevation servosystem model

Rescaling:

Thus units of $\{u_k\}$ are in binary thus any 12 bit signed integer ranging from -4095 to +4095 refers to the duty cycle of the PWM applied to the motor. The output y_k is in hundredths of degrees. To rescale we multiply b_1 and b_2 by the scaling factor:

$$\text{SCALING FACTOR} = \frac{Vcc}{2^{12}}$$

The SCALING FACTOR is= 0.003273

The azimuth servosystem model contains a zero outside the Z unit circle indicating the plant is a non-minimum zero phase system. In this instance, control system pole-placement design using zero/pole cancellation would result in an unstable response.

The elevation servosystem model contains a zero near the unit circle, pole/zero cancellation could result in a marginally stable system.

[3.14] The System Identification Program:

A copy of the C source code (SYSTIDE.C) is printed in the appendix, section.7.4B. The program implements 8 different identification techniques, each of which is performed by a subroutine (function). The subroutines are written for general use can can be applied to other identification problems.

(a) Identification Functions:

Due to the similarity in model structure, once a subroutine for one model is developed, it can be easily modified to implement a different identification scheme.

RLSE():	Recursive Least Squares Estimation
ERLSE():	Extended Recursive Least Squares Estimation
LSE():	Least Squares Estimation (non recursive)
RLSE6():	Recursive Least Squares Estimation 6 parameters
RLSE3():	Recursive Least Squares Estimation 3 parameters
RLSEO():	Recursive Least Squares Estimation+Offset
ERLSEO()	Extended Recursive Least Squares Estimation+Offset
LSEO():	Least Squares Estimation (non recursive)+Offset

(b) Matrix Functions:

The non-recursive least squares requires a matrix inversion (section 3.5), thus a matrix inversion routine was required. Matrix inversion uses Gauss elimination and scaled partial pivoting to generate an upper triangular matrix, it is then solved by back substitution. Returns false if the matrix cannot be inverted (singular matrix).

MatrixInvert()	Invert a square matrix.
MatrixMultiply():	Multiply two matrices of any given size.

(c) Graphics Functions:

Enable the results to be plotted on the IBM-PC screen in VGA mode, and alternatively plotted on an HP7475 plotter using HPGL command language.

GraphInitLinear():	Initialize the graphics screen.
GraphPlotLinear()	Plot an array of points on the screen
HPGLInitLinear():	Initialize the HP plotter (HP-GL command language)
HPGLPlotLinear():	Plot an array of points on the HP plotter.

(d) Model verification:

The following graphs 3.5A,B,C,D compare the closed loop step response of the azimuth servosystem model with experimental results at different controller gains Kc using proportional control.

Graph 3.5A Shows the closed loop azimuth servosystem model step response with the controller gain Kc=20. The red curve is the model, the green curve is the measured response.

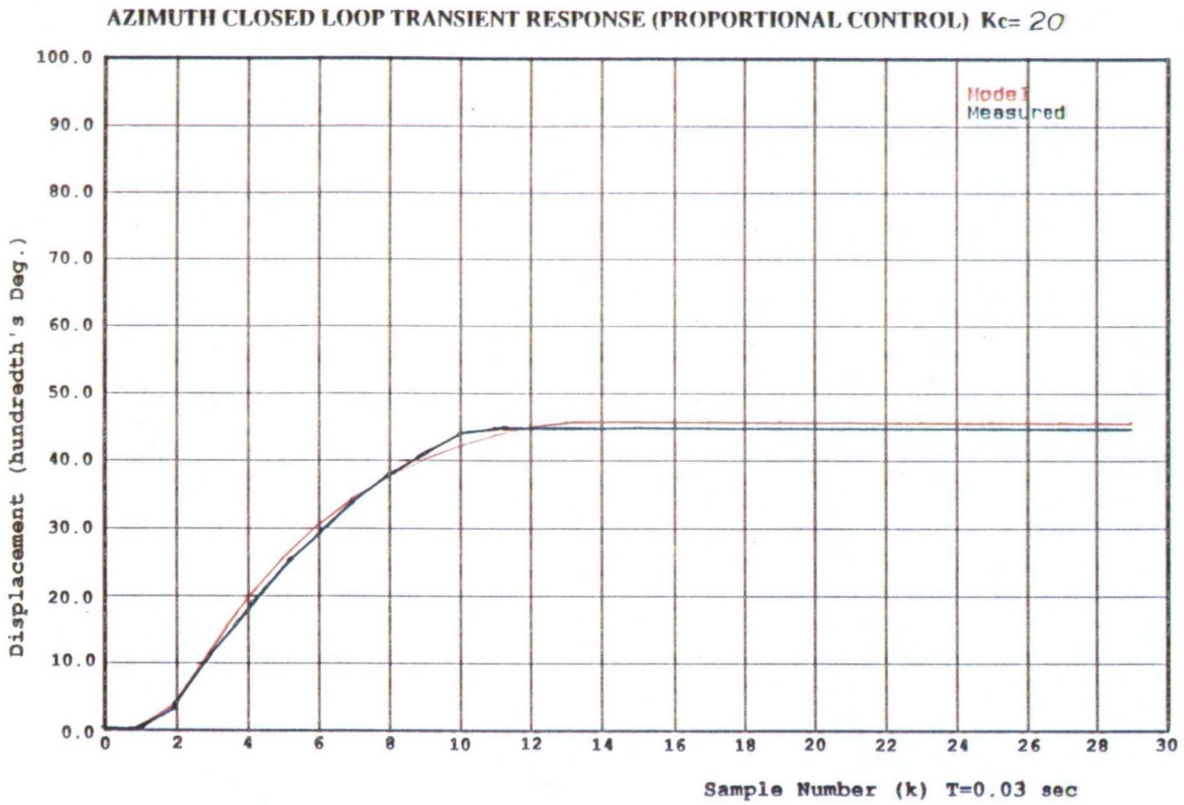
Graph 3.5B Shows the closed loop azimuth servosystem model step response with the controller gain Kc=30. The red curve is the model, the green curve is the measured response.

Graph 3.5C Shows the closed loop azimuth servosystem model step response with the controller gain Kc=40. The red curve is the model, the green curve is the measured response.

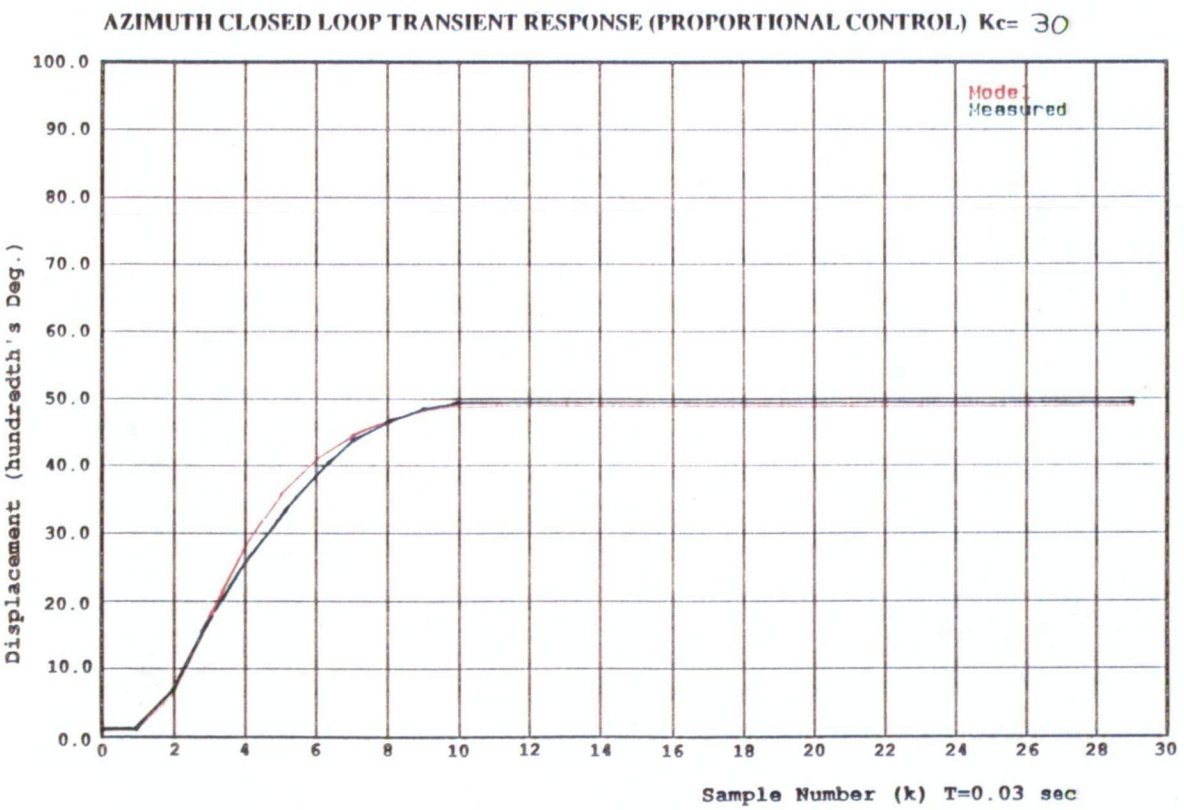
Graph 3.5D Shows the closed loop azimuth servosystem model step response with the controller gain Kc=60. The red curve is the model, the green curve is the measured response.

The model for the azimuth servosystem agrees well with the actual measured response. This model will later be used for control system design and analysis.

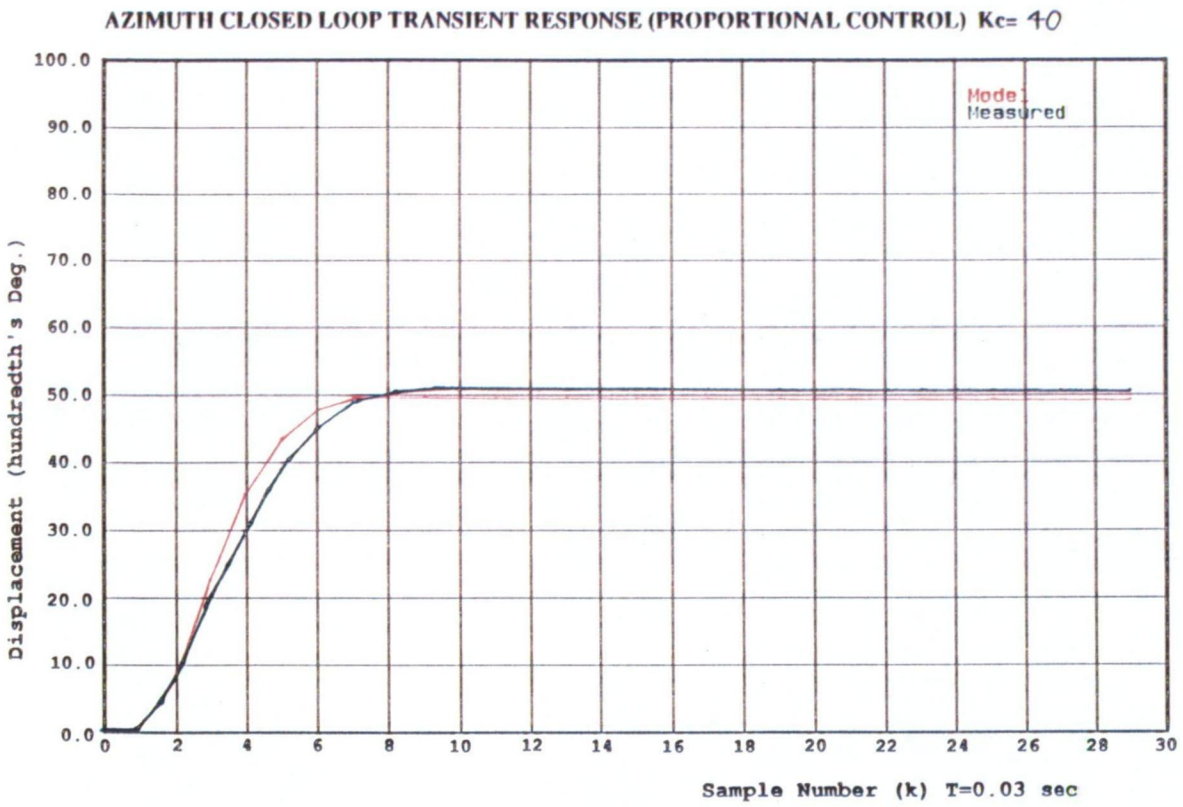
Graph 3.5A: Closed Loop Azimuth Step Response, controller gain Kc=20



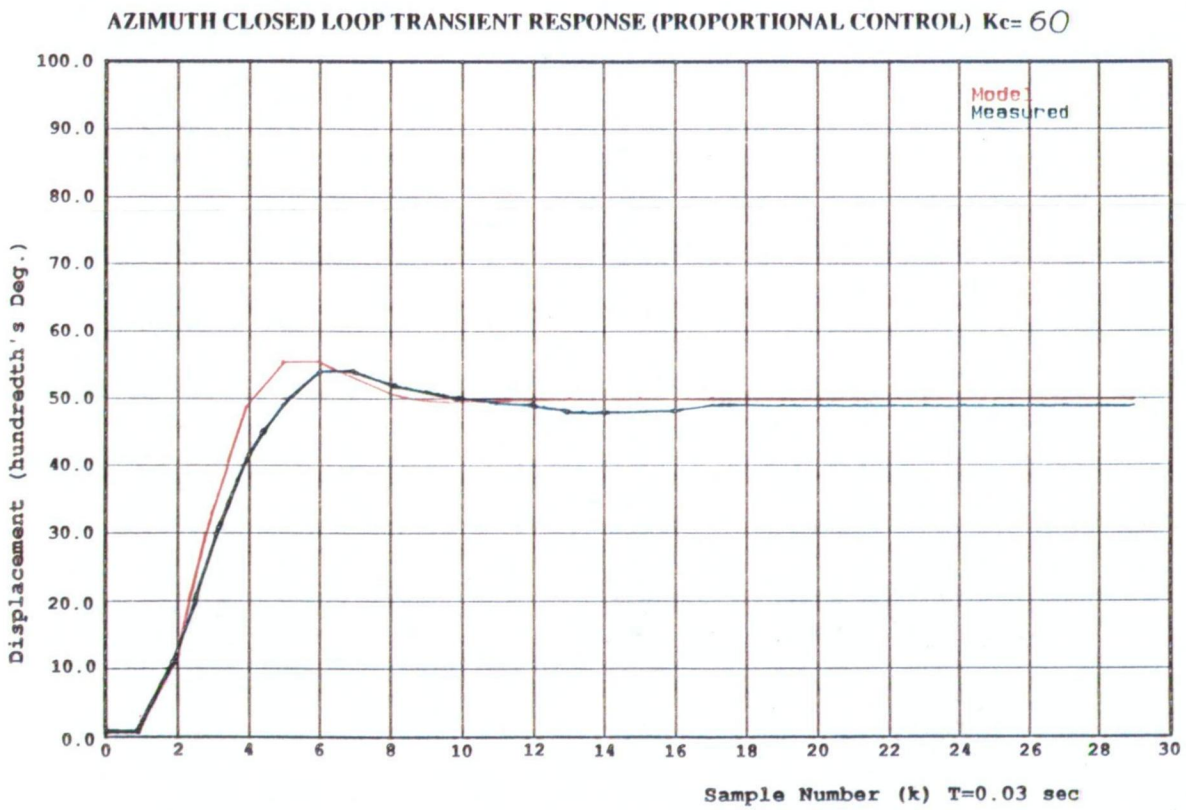
Graph 3.5B: Closed Loop Azimuth Step Response, controller gain Kc=30



Graph 3.5C: Closed Loop Azimuth Step Response, controller gain Kc=40



Graph 3.5D: Closed Loop Azimuth Step Response, controller gain Kc=60



[3.15] Chapter Summary:

In this chapter, the identification aspects of the project have been investigated. We have looked at a number of different model structures, higher order models, reduced coefficient models, with the inclusion of linear noise models, input offsets, constant trace and forgetting factor options. The results indicated that convergence is usually rapid, generally within the first 30 samples (from a total of 300 samples for each identification). Additionally, model coefficients all converge to the same values for the different model.

We were able to investigate the different model structures because the algorithm is similar in each case, and after coding the first algorithm (RLSE), it was relatively straightforward to modify to a different model and identification structure.

All the identification programs were tested and verified prior to applying on actual measurement data. Testing and verification was carried out on a simulated model with known coefficients, the program first generates a random input data sequence, the output sequence for the known model is then computed. The identification program is applied on the input and output data set and the model coefficients estimated are compared and shown to be identical with those of the known simulated model.

It would be impossible to accurately fully model the servosystem because of the presence of non-linearities, but the four parameter model was found to give reasonable accuracy (small error squared) and both recursive and non-recursive least squares method gave similar results regardless of the forgetting factor used. This indicates that the model coefficients do not change with time.

Changing the model structure by including an additional pole and zero (higher order model) did not improve the accuracy (error), similarily by reducing the order of the model by pre-fixing one pole at z=1 also had no improvement over the model accuracy.

However, the addition of an input offset term in the elevation servosystem model had resulted in a vast improvement (the error was reduced by a factor of x20), this is expected becaused the centre of gravity of the instrument generates a net torque along the elevation plane. The addition of an input offset term in the azimuth servosystem model made only slight improvement, this is probably due to non-linearities or differences in the mechanics of the servomotor operating in forward / reverse mode.

This difference may be due to friction in the worm-drive component of the servomotor, for example friction in forward rotation may not be the same as friction in reverse rotation. Because the input offset in the azimuth model is small, it will be ignored.

Chapter-4

Controller Design:

[4.1] Chapter Outline:

This chapter is subdivided into three parts as follows:

[1] Initially, the measurement model of the anglescan tracking system is developed and included as part of the feedback loop, this is found to improve the accuracy of the overall model when compared with the actual system. The measurement model can be derived from simple geometry. This model is linearized and transformed into z domain prior to being applied to the design and simulation of the controller. The non-linear model can however still be used during transient step response simulation.

[2] Simulate a closed loop proportional controller utilizing the current azimuth servosystem model developed in chapter-3 (system identification), simulate the controller with different gains. The results are then compared with actual measured data from the anglescan tracking system. This will enable the model of the closed loop proportional controller to be verified. Furthermore, the response obtained from the proportional controller model can be used as a reference to compare with the performance of more complex controller structures.

[3] A closed loop controller using the pole placement design (PPD) is implemented and simulated on computer. We compare transient response and stability with the standard proportional controller developed in [2] above.

The following programs (in addition to using the TRIP control design and simulation software package) were developed in turbo-C for controller design and simulation:

SYSTPROP.C:	Design and analysis of the Proportional Controller.
SYSTLQR.C	Design and analysis of the Linear Quadratic Controller.
SYSTPPD.C	Design and analysis of the Pole-Placement Controller.

A copy the C source code listing for each of the above programs is provided in the appendix, chapters 7.4 C, D, E respectively. Simulation using linear quadratic control was also investigated briefly, but results are not available in the thesis.

[4.2] Experimental Setup:

The following list is a brief summary of the objectives and tracking problems which the controller must overcome:

(a) Objectives:

[1] To enable "smooth" tracking and overcome deadband.
[2] To minimize steady state positioning error to step input.
[3] To minimize tracking error whilst tracking a moving target.
[4] To retain robustness and stability properties for changing (system) dynamics.

(b) Method of Design and Analysis:

As mentioned earlier, the design and analysis will proceed as follows:

[1] Develop the measurement model.
[2] Apply proportional control to verify correctness of closed loop model, this is easy to check with the experimental data.
[3] Apply pole-zero placement design - compare response with proportional control.

Only two controller types are evaluated: the pole placement controller, and the linear quadratic controller. Results are then compared with the response obtained from proportional control. The analysis will also include the measurement model which as will be seen later has the effect of de-stabilizing the control system. The model developed in section 4.3 is shown to be non-linear.

(c) Experimental Setup

Due to the limited time involved, the work in this chapter will be applied only to the azimuth servosystem, although the techniques apply equally well to the elevation servo system.

The illustration below (Fig.4.1) shows the experimental set up used for determining tracking system performance of tracking a moving target with varying velocity along both the azimuth and elevation axes. This method was adopted from a paper by G. Bayer [6.1] which uses a similar technique to determine the performance of an automatic tracking theodolite system.

We will initially compare different controller structures only from transient response to a step input by simulation and subsequently test the controller on a moving target such as illustrated in Fig.4.1 below:

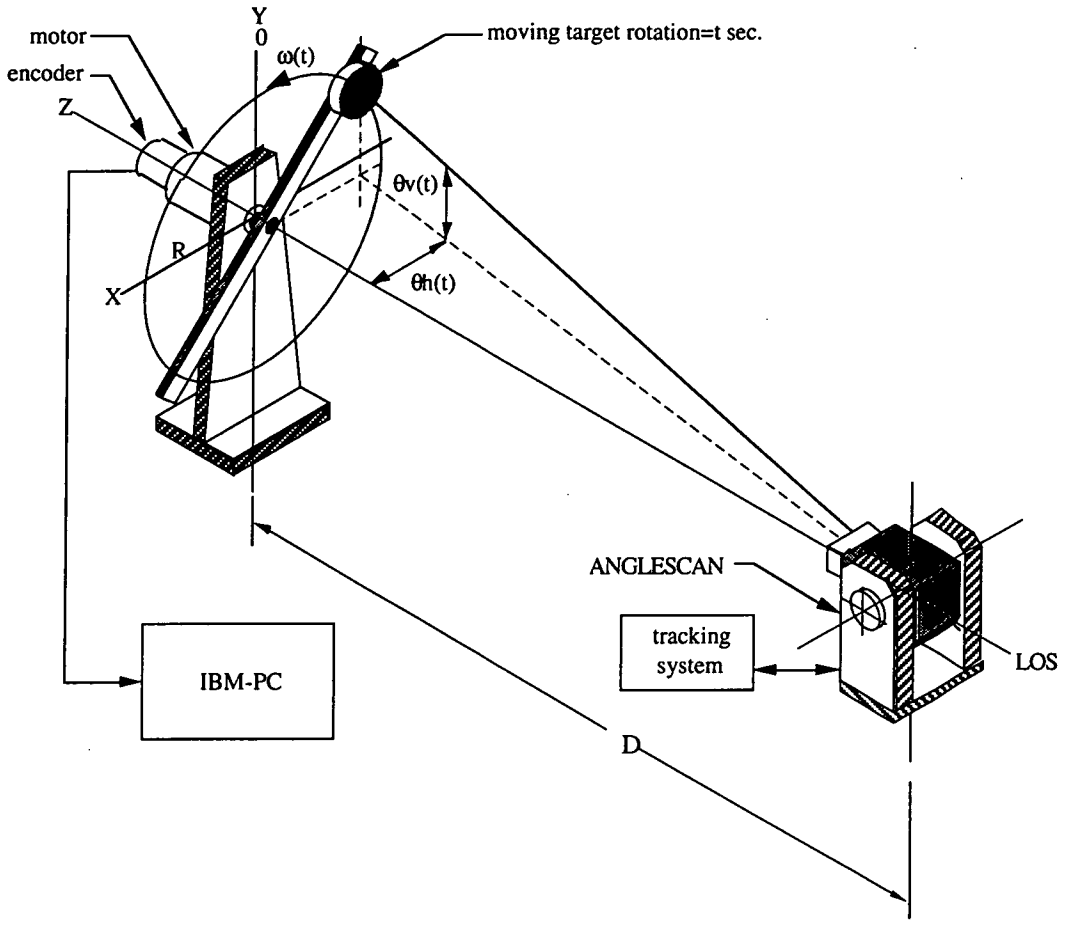


FIG.4.1 Tracking Performance Setup and Simulation

Refering to Fig.4.1, a retro reflecting mirror is placed on a rotating arm driven with a variable speed DC motor and its angular displacement measured with a shaft encoder. The distance between centres from the anglescan tracking system and the rotating arm is D, the radius of rotation is R, the angular velocity is $\omega(t) = 2\pi/T$, the instantaneous azimuth and elevation angles are $\theta h(t)$ and $\theta v(t)$ respectively. Measurements from the target mirror and anglescan system are input into a computer which computes the instantaneous azimuth and elevation errors.

Note that we assume that the control of the vertical and horizontal axes is realized as two completely separate processes. The two controllers must eliminate the influence of any mutual coupling.

We simultaneously measure the angular displacement of the moving target and the anglescan system and compare the results.

[4.3] The Anglescan Measurement Model:

(a) Introduction:

The anglescan measurement model is simply a model of the measurement process involved in obtaining the azimuth and elevation angles. Generally, the measurement of any parameter in a system is an instantaneous process occouring at each sampling instance. However in this case, because the measurement is not instantaneous but over the entire sampling period, we cannot assume that the value represented at the end of the sampling instance to be an accurate indication of the actual angle, but only an estimate.

As will be seen later, the introduction of a measurement model M(z) has a twofold effect on the closed loop system: the first is to improve the accuracy of the servo system model compared with the measured response, the second is a de-stabilizing effect upon the closed loop control system.

(b) Model Description:

Figure 4.2 below illustrates the entire system with the Measurement model M(z) included as part of the feedback path (shown inside the semidotted rectangle). The following nomenclature will be adhered to throughout this chapter (refer also to appendix 7.1B):

Gc(z)	Controller Dynamics in the z domain (implemented in software).
U(z)	Controller output (represented by a 12 bit signed binary number)
H(z)	Plant dynamics obtained in chapter 3 from system identification
$\theta c(z)$	Angular displacement of motor shaft output (azimuth or elevation), or $\theta c(k)$ in discrete time at sample k.
$\theta r(z)$	Reference angular displacement (azimuth or elevation), or $\theta r(k)$ in discrete time.
E(z)	Angular displacement error between the reference and the system output, or e(k) in discrete time at sample k.
M(z)	Measurement model dynamics (to be developed in section 4.3).
$\hat{E}(z)$	Estimated angular displacement error from measurement model, or $\hat{e}(k)$ in discrete time at sample k.
$\alpha_1,\alpha_2,\alpha_3,\alpha_4$	Four measurements obtained from the anglescan optical system (refer to Fig.1.11A,B,C,D) from which the azimuth and elevation angles are calculated.
G(z)	Is the product of the measurement model and plant dynamics G(z)=M(z).H(z)

Without the measurement model, the feedback path would simply be: $\hat{E}(z) = E(z)$ A detailed description of the measurement system has already been provided in section 1.3.

The measurement model can be developed from the geometrical properties of the scanning laser beams, the location and velocity of the target (mirror).

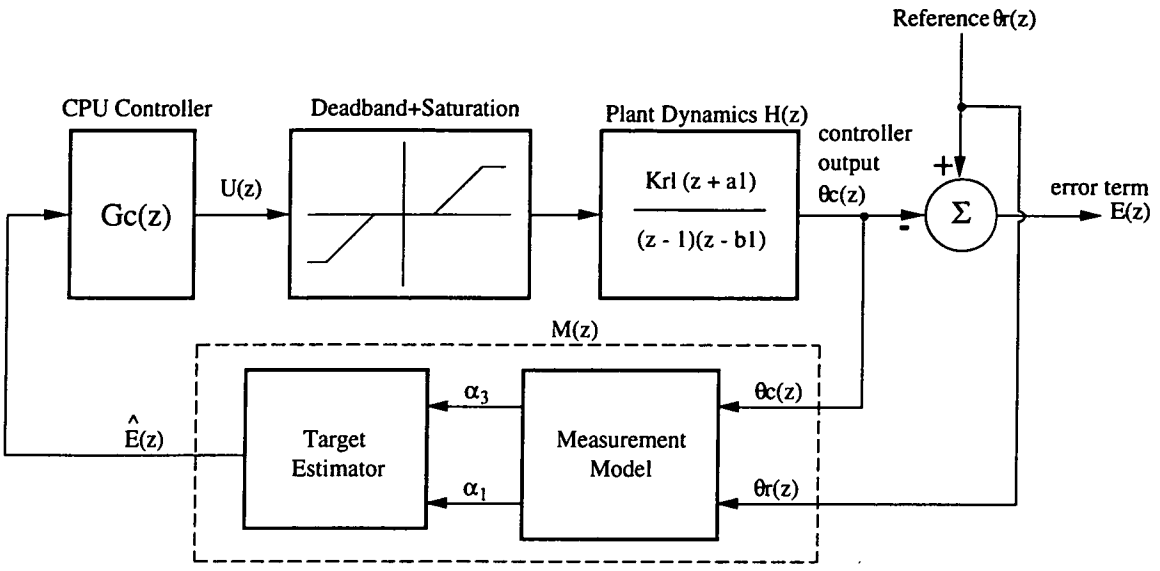


Fig.4.2 Closed Loop Control System with Measurement Model

Note that the parameters θr(z) and θc(z) are not available to the anglescan computer, however only the E(z) term is available.

The measurement model can be described accurately by a non-linear function which will be derived in this section. The target estimator is essentially the set of equations provided below, the estimated error is given by:

Measurement model error:

Described in discrete time at sampling instance k by the following equation:

$$\hat{E}(k) = M\big(\theta r(k), \theta c(k)\big) \tag{4.2}$$

where M() represents a non linear measurement process and target estimator equations. The true error if no measurement process were present is the instantaneous difference, which would simply be given by:

True error:

The true error between the controller output and the reference is given by the equation 4.3 below:

$$E(k) = \theta r(k) - \theta c(k) \tag{4.3}$$

(c) Derivation of Measurement Model:

The servosystem model obtained from system identification in chapter 3 does not include the measurement process. The measured output was taken directly from the shaft encoder $\theta c(z)$ and not the optical system, thus the error e(k) represented the true error given by equation 4.3 and not the error $\hat{e}(k)$ from the measurement optics which is given by equation 4.2.

During target tracking, the optical system must be used to determine the relative position of the target. It is necessary to develop a measurement model relating the true error e(k) to the estimated error $\hat{e}(k)$ from the measurement process in order to design a controller for target tracking applications.

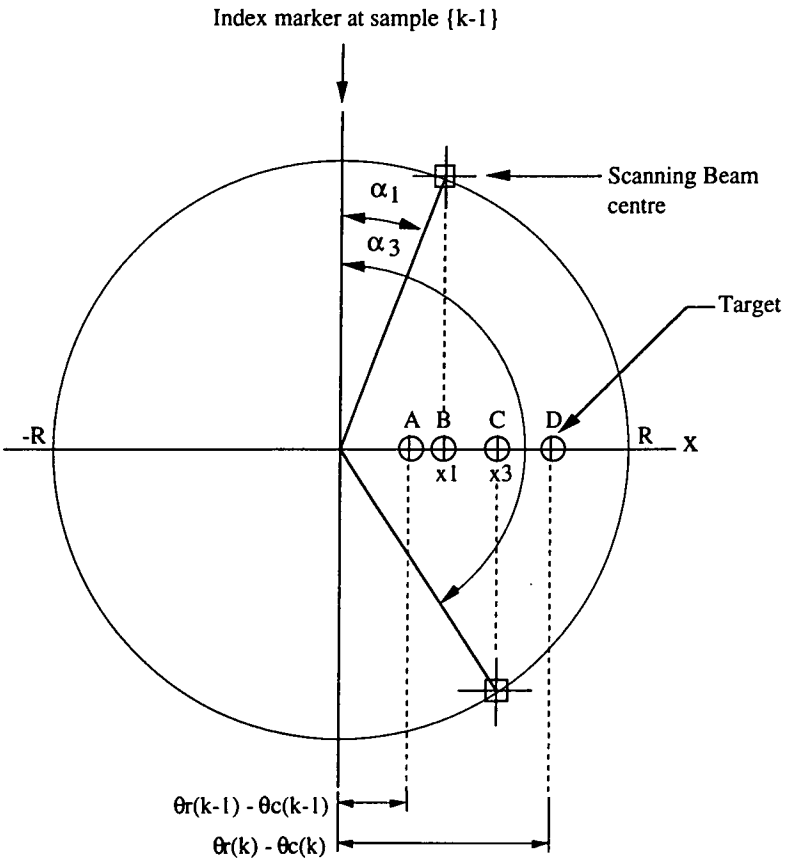


Fig.4.3 Apparent Target Motion While Scanning

To develop the measurement model, we refer to Fig.4.3, at the commencement of the sample, the target is located at point **A**, the laser cross is at the top dead centre at this time. By the time the cross rotates by a displacement of α_1 the target would have moved to point **B**. Then the cross further rotates by a distance α_3 and the target has now moved to point **C**. Finally at the end of the sample (top dead centre again) the target would have moved to point **D**. We will now develop this concept into a set of equations, refer to fig.4.4 also:

[1] Assume that at sample {k-1} the target is at $\theta c(k-1)$, and the desired (reference) is at: $\theta r(k-1)$. Thus we have: the point A on Fig.4.3, below as the true error ie:

$$\text{POINT A:} \quad e(k-1) = \theta r(k-1) - \theta c(k-1) \tag{4.4A}$$

similarily at the next sampling instance k we have the true error as:

$$\text{POINT D:} \quad e(k) = \theta r(k) - \theta c(k) \tag{4.4B}$$

[2] Next, the rotating cross intersects the moving target at points α_1 (ie t1) and then at α_3 (ie: t3) refer to Fig.4.4, the target trajectory is assumed to be a straight line with relative velocity as the difference between the target's velocity and the instrument's tracking velocity, furthemore the relative velocity is assumed constant between samples. Thus the position of the target relative to the centre of the circle (centre of field) is from interpolation, where x1 and x3 represent the azimuth at time t1 and t3 respectively:

$$\text{POINT B:} \quad x1 = e(k-1) + V(k).t1 \tag{4.4C}$$

$$\text{POINT C:} \quad x3 = e(k-1) + V(k).t3 \tag{4.4D}$$

where t1 is the time to travel distance α_1 and t3 is the time to travel distance α_3 from the top dead centre. The relative (average) velocity term is calculated between samples and given by:

$$V(k) = \frac{e(k) - e(k-1)}{Ts} \tag{4.5}$$

Where Ts=sample time. The two points of intersection t1 and t3 are found by the intersection of the straight line (target motion) and the sinuosoid function (of the rotating cross) as shown in fig.4.4, where t is in the range: $0<t<Ts$

straight line equation: $x(t) = e(k-1) + V(k).t$ (4.6A)

sinuosoid equation: $x(t) = R.\sin(2\pi.t/Ts)$ (4.6B)

the points of intersection of t1 and t3 are at:

$$e(k-1) + V(k).t1 = R.\sin(2\pi.t1/Ts) \quad (4.7A)$$
$$e(k-1) + V(k).t3 = R.\sin(2\pi.t3/Ts) \quad (4.7B)$$

We solve for the values of t1 and t3 by applying Newton's method recursively: define f(t) and assume to be a continuously differentiable function:

$$f(t) = e(k-1) + V(k).t - R.\sin(2\pi.t/Ts) \quad (4.8)$$

the derivative with respect to time is:

$$f'(t) = V(k) - \frac{2\pi R}{Ts}.\cos\left[\frac{2\pi.t}{Ts}\right] \quad (4.9)$$

because the solution to the above equation can only be found by iteration a better estimate of t can be found by the recursive form:

$$t(n+1) = t(n) - \frac{f[t(n)]}{f'[t(n)]} \quad (4.10)$$

The method is found to converge very rapidally, generally requiring three to four iterations for the solution to reach the necessary accuracy.

The procedure **"TargetMeasurementModel()"** in the program: SYSTPROP.C (appendix 7.4C) implements the above estimation algorithm.

(d) Summary:

Refering to Fig.4.4, we estimate the values of α_1 and α_3 (the expected values obtained by the anglescan tracking system) by solving recursively the equation for the two points of intersection for t1 and t3 by using the Newton iteration described in eqn 4.10:

$$e(k-1) + V(k).t = R.\sin(2\pi.t/Ts) \tag{4.11}$$

where: $e(k-1) = \theta r(k-1) - \theta c(k-1)$

$e(k) = \theta r(k) - \theta c(k)$

$V(k) = (e(k) - e(k-1))/Ts$

solve for t1 and t3 from 4.11 above, we then find the next estimate of $\hat{e}(k)$ from the target estimator function:

$$\hat{e}(k) = \frac{1}{2}(x1 + x3) \tag{4.12}$$

where: $x1 = R.\sin(2\pi.t1/Ts)$

$x3 = R.\sin(2\pi.t3/Ts)$

Note that the estimate $\hat{e}(k)$ is estimated from the simulation model, this is equivalent to the anglescan measurement system output. The values e(k) and e(k-1) are also calculated from the simulation model but are not available as measurements in the actual anglescan system, this is why the parameter $\hat{e}(k)$ must be used in the simulation model.

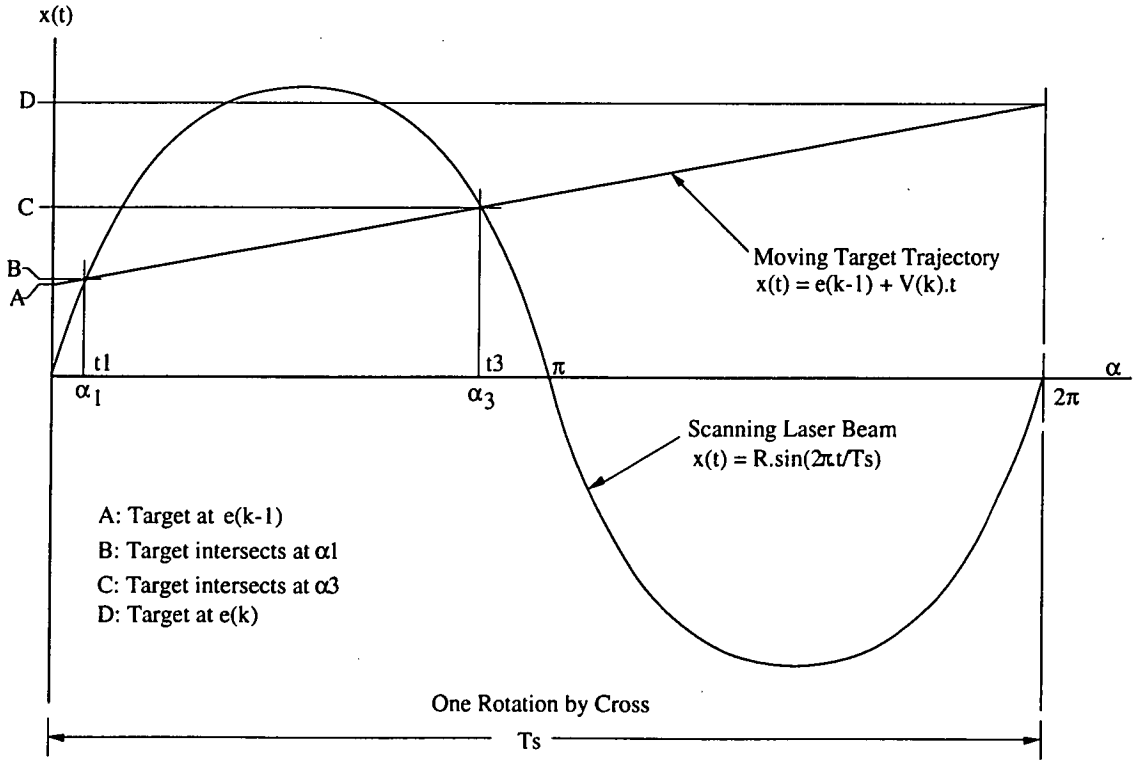


Fig.4.4 Laser Beam Intersecting the Moving Target at $\alpha 1$ and $\alpha 3$

(e) Measurement Model Linearization:

The measurement process described above can be linearized by using least squares identification technique developed previously in chapter 3. Consider the following plant model and measurement process shown in fig.4.5.
The measurement model M(z) is a linearized approximation to the non-linear measurement process developed earlier. It is assumed that M(z) has the following transfer function:

$$M(z) = \frac{b1.z^{-1} + b2.z^{-2}}{1 - a1.z^{-1} - a2.z^{-2}} \qquad (4.13)$$

Fig.4.5 below illustrates the overall control system with the measurement process denoted by the discrete transfer function: M(z):

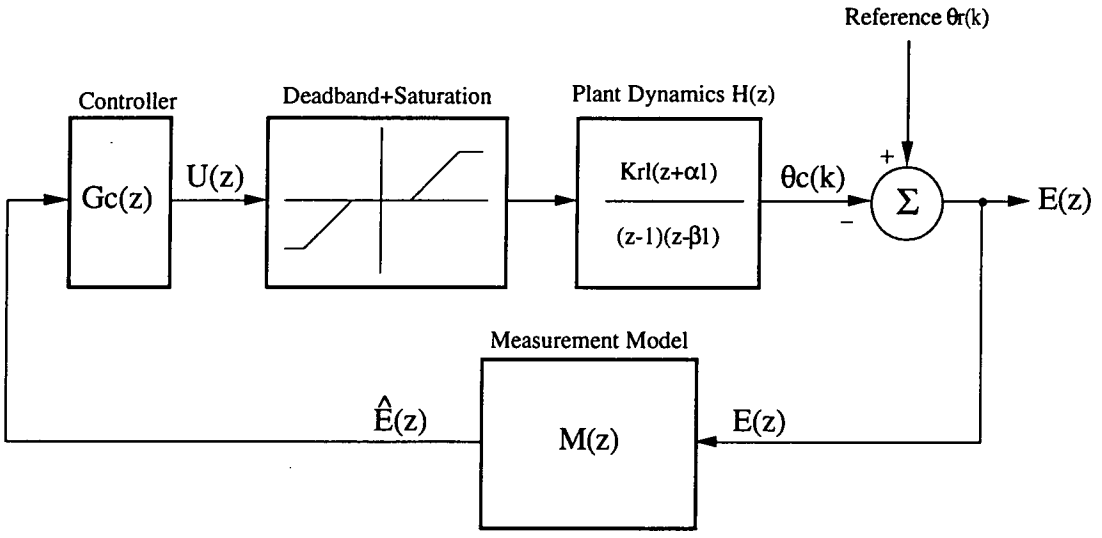


Fig.4.5 Closed Loop Control System with Measurement Model

where M(z) is the transfer function of the measurement process in z domain thus:

$$\frac{\hat{E}(z)}{E(z)} = M(z)$$

In order to obtain the b1,b2,a1,a2 coefficients, we require to generate a random PRBS input sequence e(k) and then calculate the output sequence from the non-linear model: $\hat{e}(k)$, the input/output data sequence is subsequently used to solve for the above coefficients by the application of least squares. Least square algorithms developed in chapter 3 can be used.

Refering to fig.4.6, we generate a random set of input data e(k) and calculate from the non-linear measurement model the output sequence $\hat{e}(k)$, and then apply the system identification (least squares) algorithm to the data set and solve for the four unknown coefficients a1,a2,b1,b2.

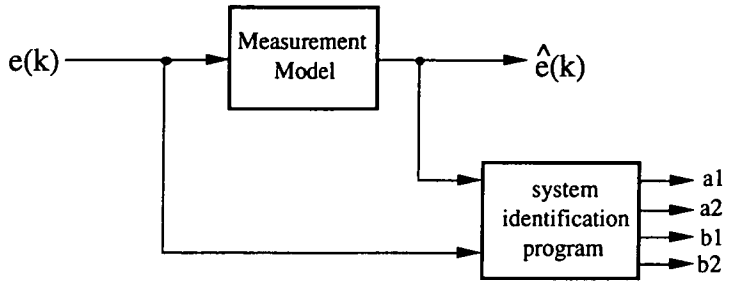


Fig.4.6 Measurement Model Linearization by PRBS input

where the sequence in Fig.4.25 e(k) is a random PRBS input, and $\hat{e}(k)$, is calculated from the non-linear model described by equations 4.11 and 4.12

Simulation results:

The identification program (not listed in the appendix) IDENMEAS.C is used to implement the first identification scheme, giving the following M(z) model results:

a1= 0.429753
a2=-0.193080
b1= 0.607255
b2= 0.153779

thus giving the following model structure:

$$M(z) = \frac{0.607255 + 0.153779.z^{-1}}{1 - 0.429753.z^{-1} + 0.193080.z^{-2}} \qquad (4.14)$$

The following graphs (4.2A,B,C,D) show the closed loop step response and PRBS response comparing the non-linear model with the linearized z-domain model. The z domain model agrees well with the time domain model.

The first three graphs show the step response of the closed loop system illustrated by Fig.4.5 with the measurement model given by equation 4.14 set with gain k1=10,20,30 respectively. The last graph is the PRBS response of the model in time and z domain only as depicted by fig.4.6.

(f) Measurement Errors:

There are two kinds of errors which can occur because of the motion of the target across the field of view. These errors have being found to occur during measurements and testing. The two error conditions are (1) overcount errors, (2) undercount errors. Consider the condition for the azimuth measurements only:

Overcount:

This condition occurs during high slew rate target tracking, it means that the rotating laser cross intercepts the target more than twice (this case is three times) producing an overcount condition. The target measurement hardware is capable of detecting this condition. When this condition occurs, an error is flagged by the measurement logic to the computer.

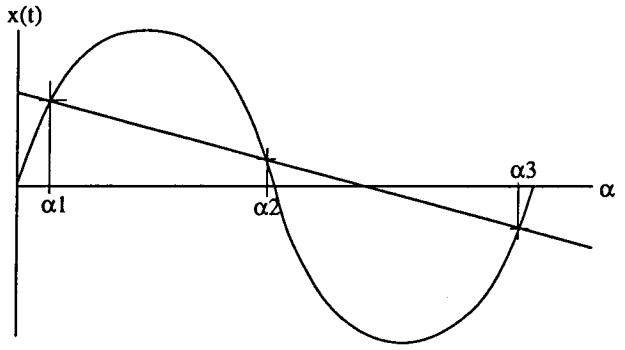


Fig.4.8A Overcount condition

Undercount:

This condition occurs during target tracking when the target is moving in the opposite direction. When this happens, only one measurement is obtained instead of two giving an undercount condition. The target measurement hardware is capable of detecting this condition also.

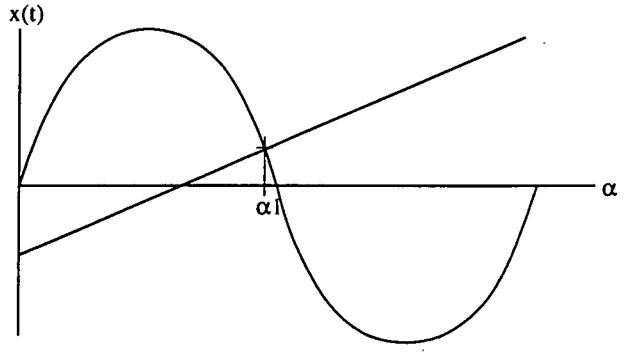


Fig.4.8B Undercount condition

Again, when this condition occurs, an error is flagged by the measurement logic to the computer.
These two error conditions can never occur if the target is stationary with respect to the field of view of the instrument.

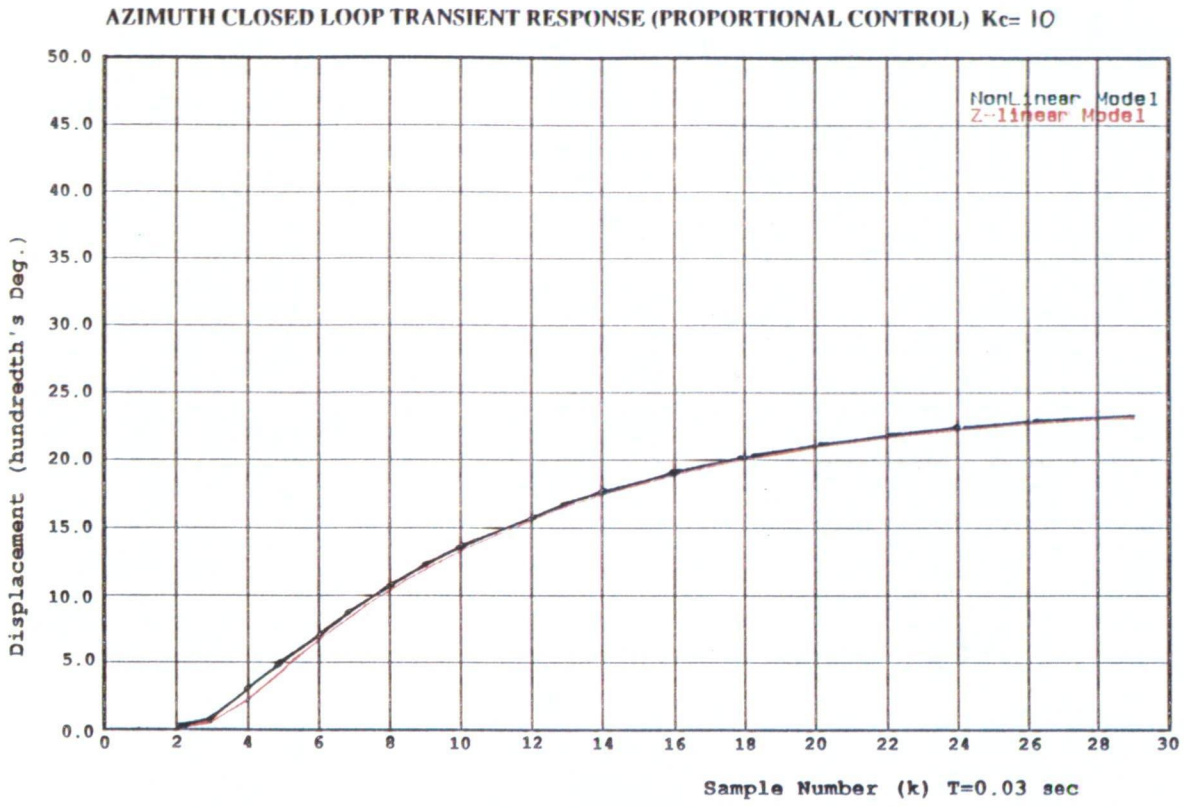
Summary of graphs:

Graphs on the following pages illustrate the transient step response of the closed loop azimuth servosystem with the measurement model included in the feedback path (see fig.4.5). The two responses compare the linearized z domain model M(z) given by equation 4.14 with the time domain model given by equations 4.5 to 4.12.
Each graph represents the response for different controller gains Gc(z)=Kc being a proportional controller.

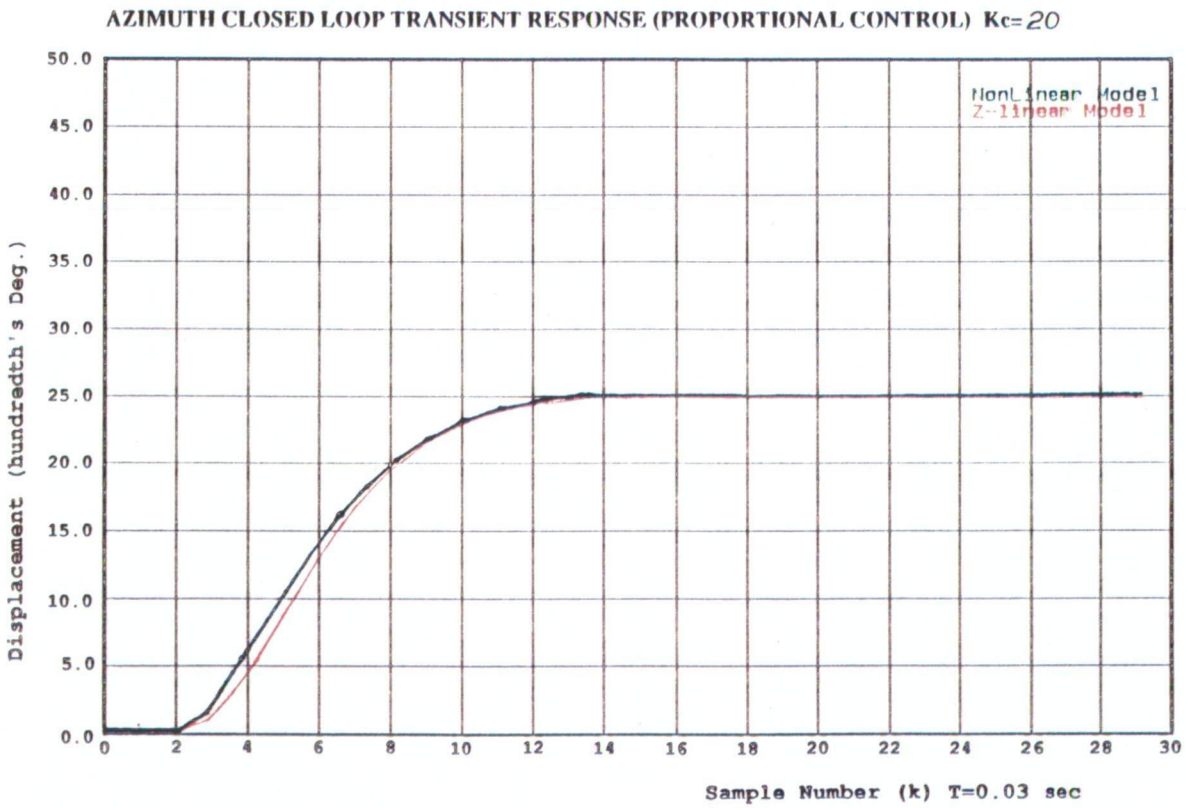
Graph 4.2A:	Is a step response with the controller gain Kc=10.
Graph 4.2B:	Is a step response with the controller gain Kc=20.
Graph 4.2C:	Is a step response with the controller gain Kc=30.
Graph 4.2D:	Is a step response with the controller gain Kc=40.
Graph 4.2E:	PRBS response of the linearized and non linear measurement model.

In all the above graphs, the green curve is the non linear measurement model, the red curve is the linearized measurement model in z domain M(z). The graphs simply compare the transient step response of the closed loop proportional controller with differing gains Kc to verify that the linearized model agrees with the non linear model.
The last graph 4.2E is the PRBS response of the linear and non linear measurement model only (removed from the loop) as shown in fig.4.6.

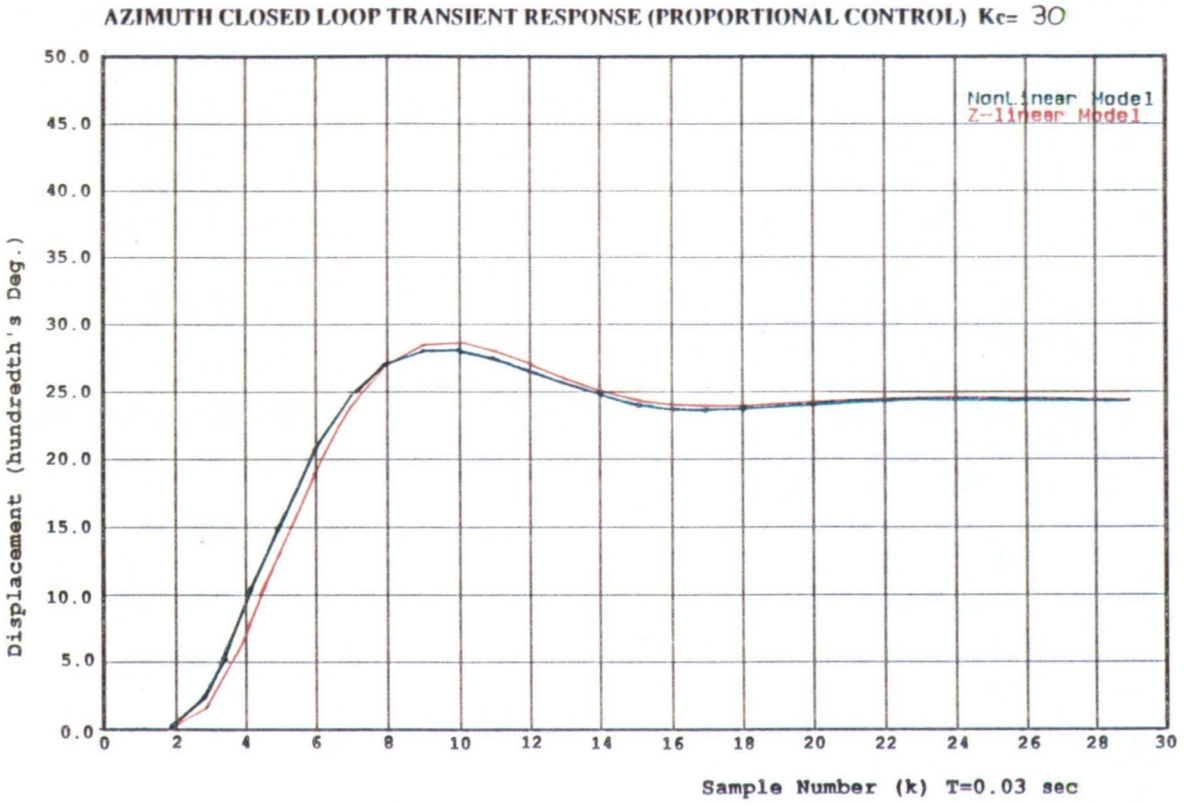
Graph.4.2A: step response of the closed loop system with the controller gain set to Kc=10



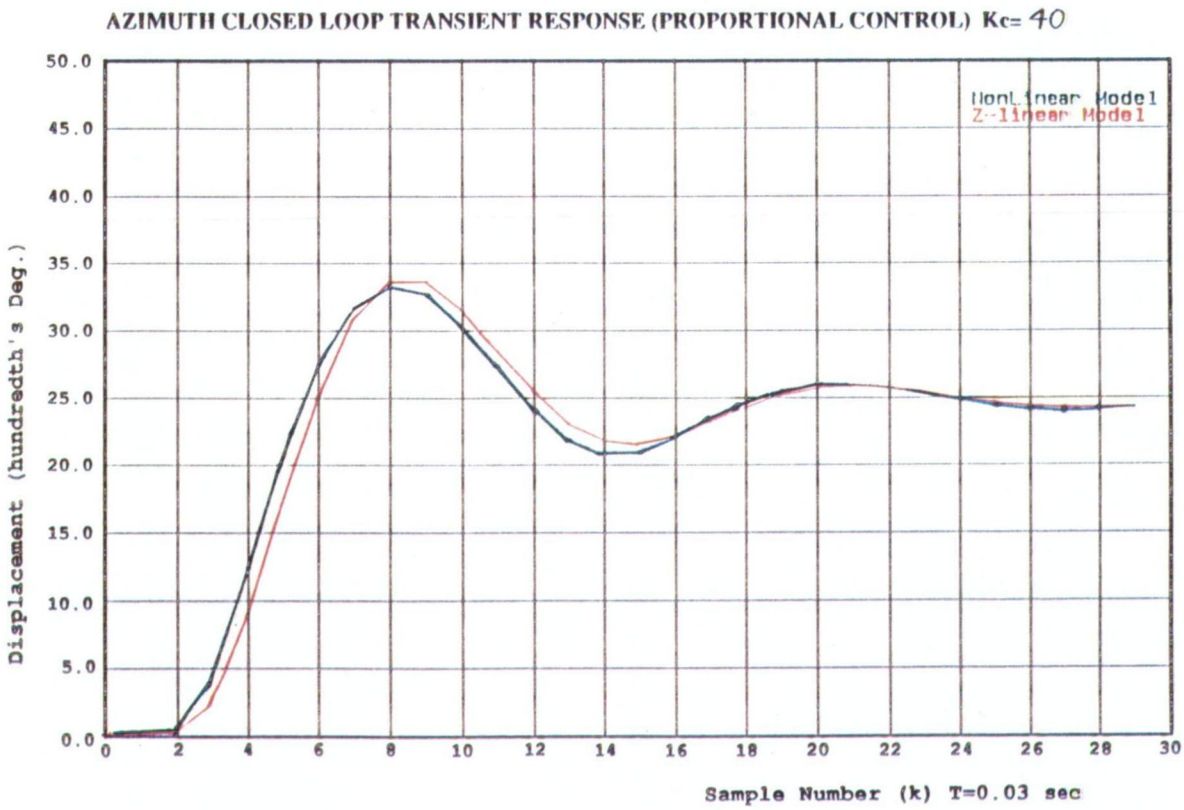
Graph.4.2B: step response of the closed loop system with the controller gain set to Kc=20



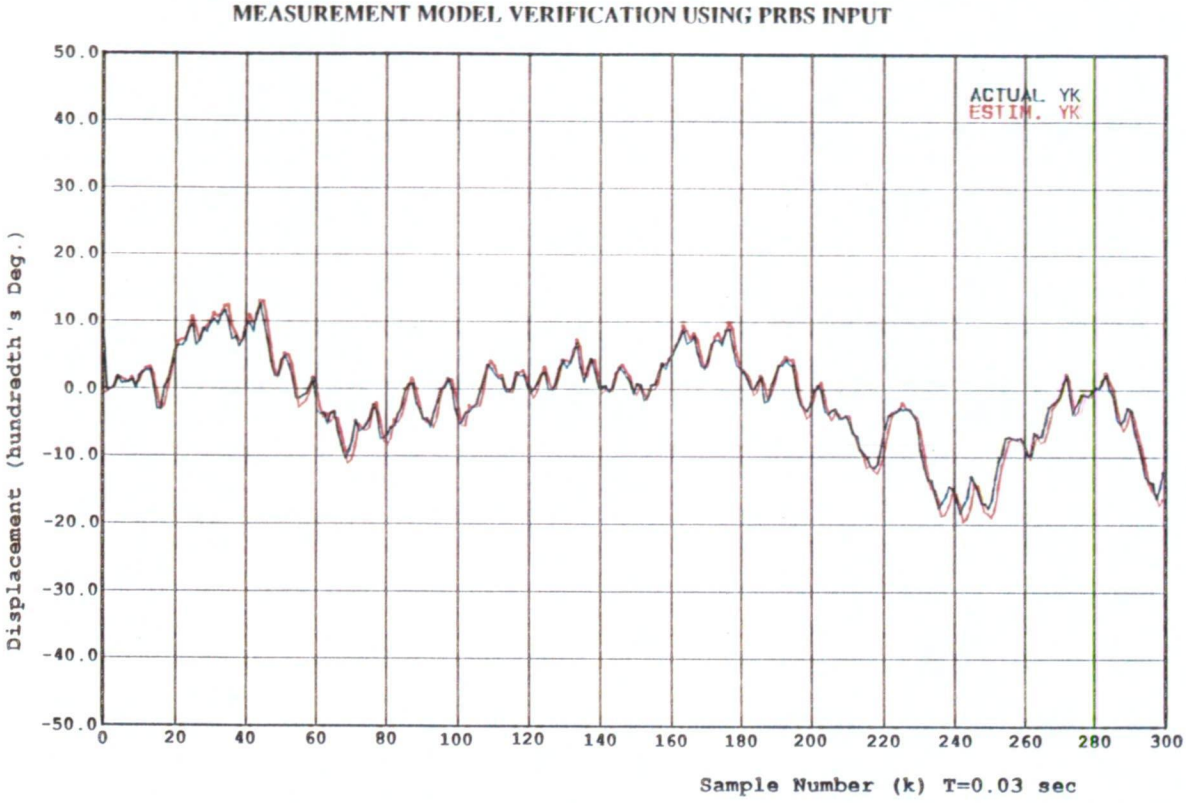
Graph.4.2C: step response of the closed loop system with the controller gain set to Kc= 30



Graph.4.2D: step response of the closed loop system with the controller gain set to Kc=40



Graph.4.2E PRBS response of linear and non linear measurement model



[4.4] The Proportional Controller:

The simplest form of closed loop feedback control is proportional control. Figure 4.9 illustrates the block diagram of the proportional controller with the measurement model included in the feedback path. Deadband and saturation are present and thus simulated in the forward path. In this section we will examine the following aspects of the anglescan tracking system using proportional control:

(a) Target acquisition dynamics.
(b) Velocity tracking error.
(c) Step response simulation and comparison with experimental results.

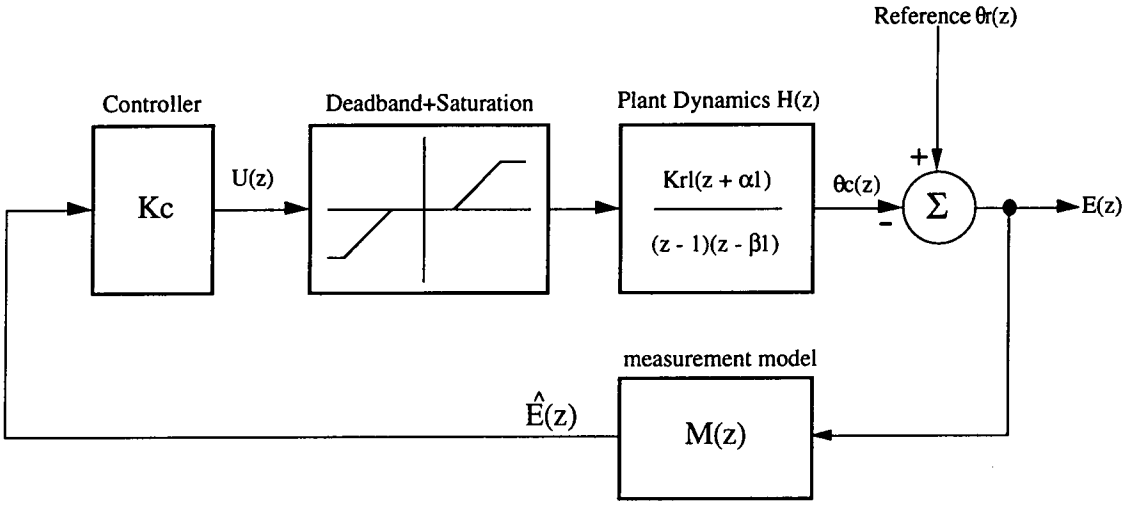


Fig.4.9 Closed Loop Control System With Proportional Control

We conduct step response measurements and compare the above azimuth model with coefficient values of: Krl=0.0036, α1=1.20, β1=0.120 at varying controller gains of Kc=30 and 40. Results are shown on the following graphs. Simulation is conducted in z-domain, sampling time Ts=0.03 seconds using the developed measurement model.

Note that the maximum size of the step must be less than the field of view of the instrument, subsequently this is subjected to noise and backlash which are considerable at this amplitudes and model verification is difficult in practice. The maximum amplitude should be less than 0.5 degrees (FOV of instrument).

Additionally, to avoid actuator saturation, the maximum gain of the controller Kc is limited to the following value:

$$Kc < \frac{bn(sat)}{e_{max}(k)} \tag{4.15}$$

where bn(sat) is the applied binary value which saturates the PWM actuator, this is = 4096, the maximum value of step $e_{max}(k)$ is 50 hundredths of degree, consequently the value of gain Kc should be less than about 80 to avoid actuator saturation at maximum amplitude.

(a) Target Acquisition Dynamics:

We simulate two aspects of target tracking: (1) maximum target velocity allowed for target aquisition, (2) steady state velocity tracking error. Graphs are shown below. It is assumed that the instrument's field of view is 1.00 degrees.

Fig.4.10 below is a simulation of the above system, with the target moving into the field of view of the instrument with velocity shown on the Y axis, as a function of controller gain. Thus if the gain Kc is set to 40, then the maximum target velocity which will enable the instrument to capture and continue tracking the moving target is 4.2 degrees/sec.

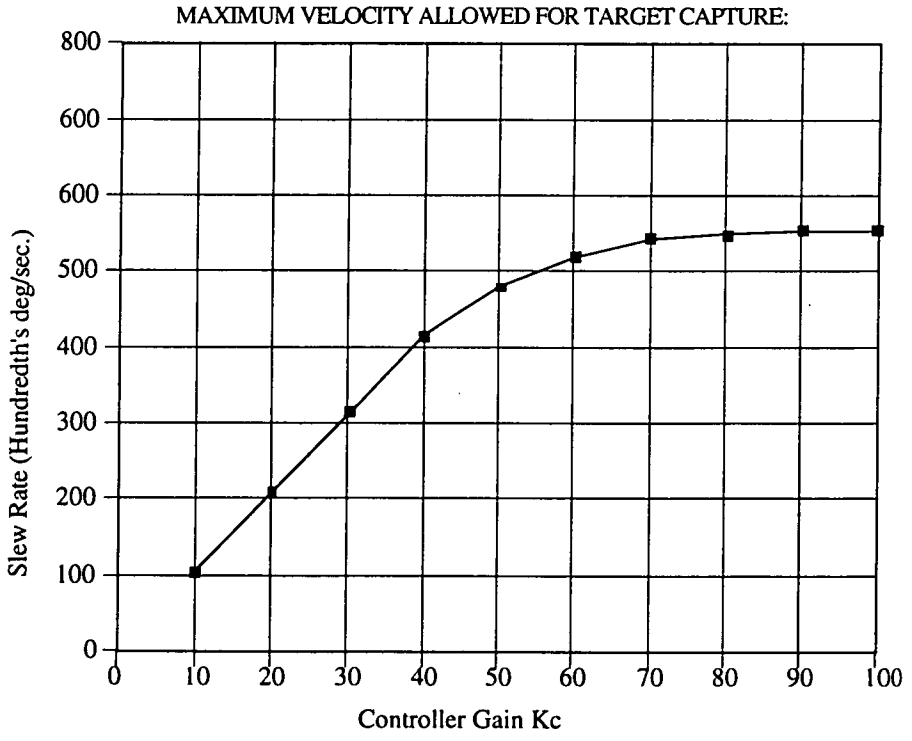


Fig.4.10: Maximum Velocity Target Acquisition

The above results were obtained by simulating a target mirror moving along the x axis just comming into the field of view of the instrument, then applying the servomotor model and the measurement model to determine the acquisition response of the tracking system.

Figure 4.10B below illustrates the target acquisition simulation. The target starts at point -θr at time t=0 and moves to point θr at time t=T. We assume uniform velocity, therefore the position of the target in continuous time is given by:

$$\theta r(t) = -\theta r + \frac{2\,\theta r}{T}.t \qquad (4.15A)$$

When the target comes into the field of view of the instrument, ie: -R<θr(t)<R, then the program begins the acquisition and tracking of the target.

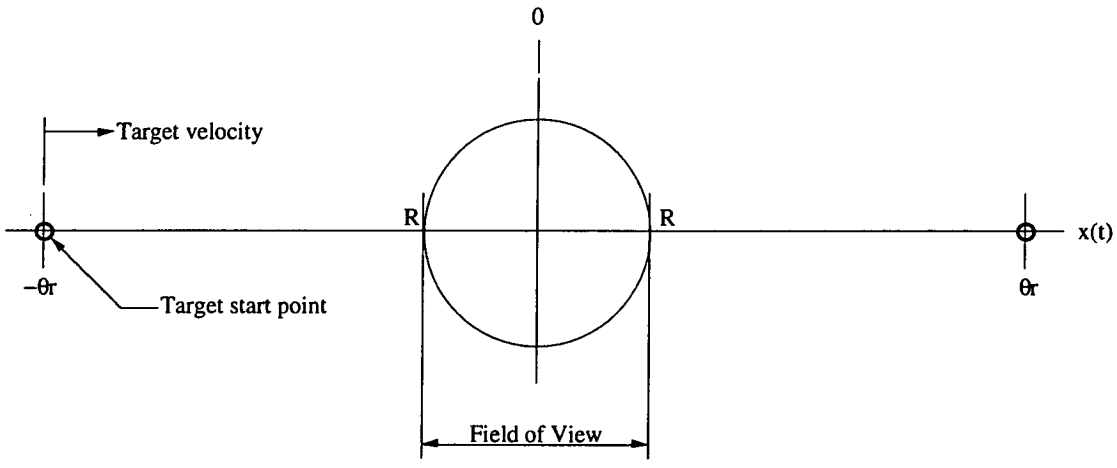


Fig.3.10B Simulation of Target acquisition

Graphs 4.3C and 4.3D show the target acquisition and tracking simulation results of the proportional controller with gains Kc=50 and Kc=100 respectively. The response compares the model-1 (red) which is the tracking system with infinite field of view, and the model-2 (blue) which is the tracking system with the field of view limited from: -R to +R. The black line is the position of the target (which is a straight line from -θr to +θr). The graph shows -θr=200 hundredths of degrees and +θr=200 hundredths of degrees. The X axis is time in seconds, and T=6 seconds. The effects of deadband and stiction are not included in the simulation.

(b) Steady State Velocity Tracking Error:

Below fig.4.11 illustrates the steady state velocity tracking error as a funtion of the controller gain Kc for the system illustrated in fig.4.9.
Calculation of the velocity tracking error is found by making the input function θr(t) a ramp (velocity function) and calculating the steady state error $e_{ss}(t)$.
Note that when the steady state error exceeds the field of view of the instrument, then tracking is lost.

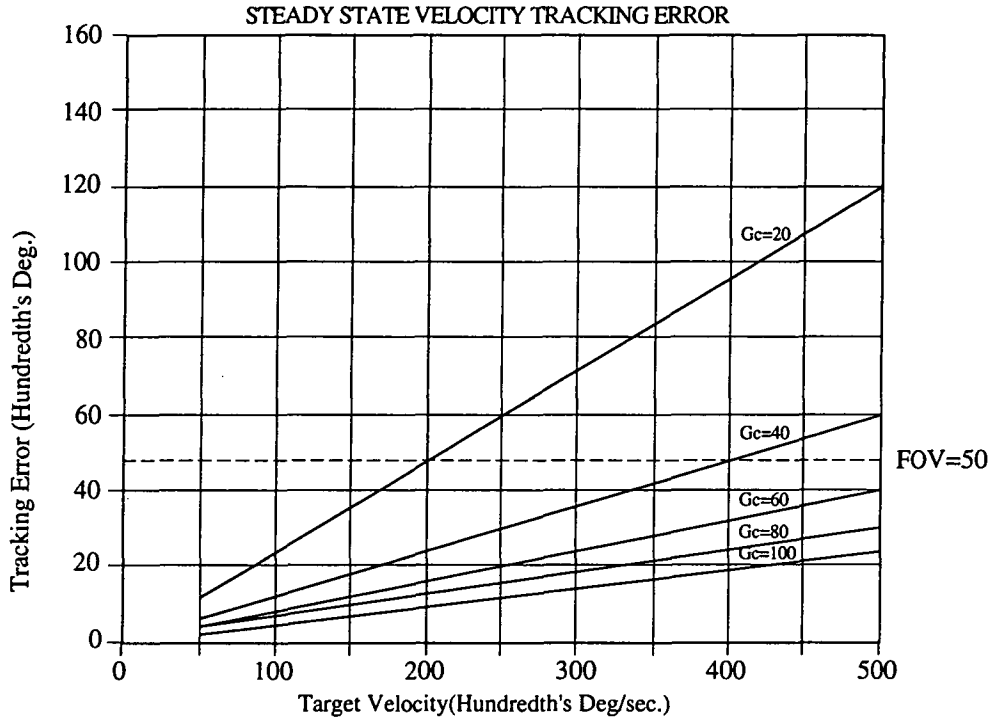


Fig.4.11: Velocity Tracking Error to Ramp Input

The steady state velocity tracking error is calculated from the final value theorem. Given that the loop gain is G(z)=Kc.H(z).M(z) and T=sampling time, the steady state velocity error is:

$$e_{ssv} = \lim_{z \to 1} \left[(z-1) \frac{Tz}{(z-1)^2} \frac{1}{1 + G(z)} \right] \qquad (4.15b)$$

(c) Step Response Simulation Results:

The figures on the following pages show transient response measurements and are compared to simulation results. The program SYSTPROP.C is used to simulate the step response with the measurement model included. (Refer to 7.4C in the appendix).
Referring to figures 4.3A and B, we can compare the theoretical model with experimental results. The results indicate that the experimental model agrees well with the measured experimental data.
Furthermore, the results also indicate that with the measurement model included, the system becomes unstable at lower controller gains, for example at Kc=30 there is an overshoot occuring. When we compare this with figure 3.5B (chapter 3) which uses an identical model but does not include the measurement model in the feedback path, we can see that there is no overshoot in this instance. In fact, the system does not begin to show an overshoot condition until the controller gain becomes Kc=60.

Simulation Program:

A copy of the proportional controller simulation program **SYSTPROP.C** is included in the appendix 7.4C. The program is able to completely simulate the model shown in figure 4.9 and produce a graphical output of the transient response on an HP plotter and graphics screen.

Graph 4.3A: Closed loop transient step response of the model (red) and the measured system (green) using proportional control at gain Kc=10. The measurement model is included in the feedback. See fig.4.5.

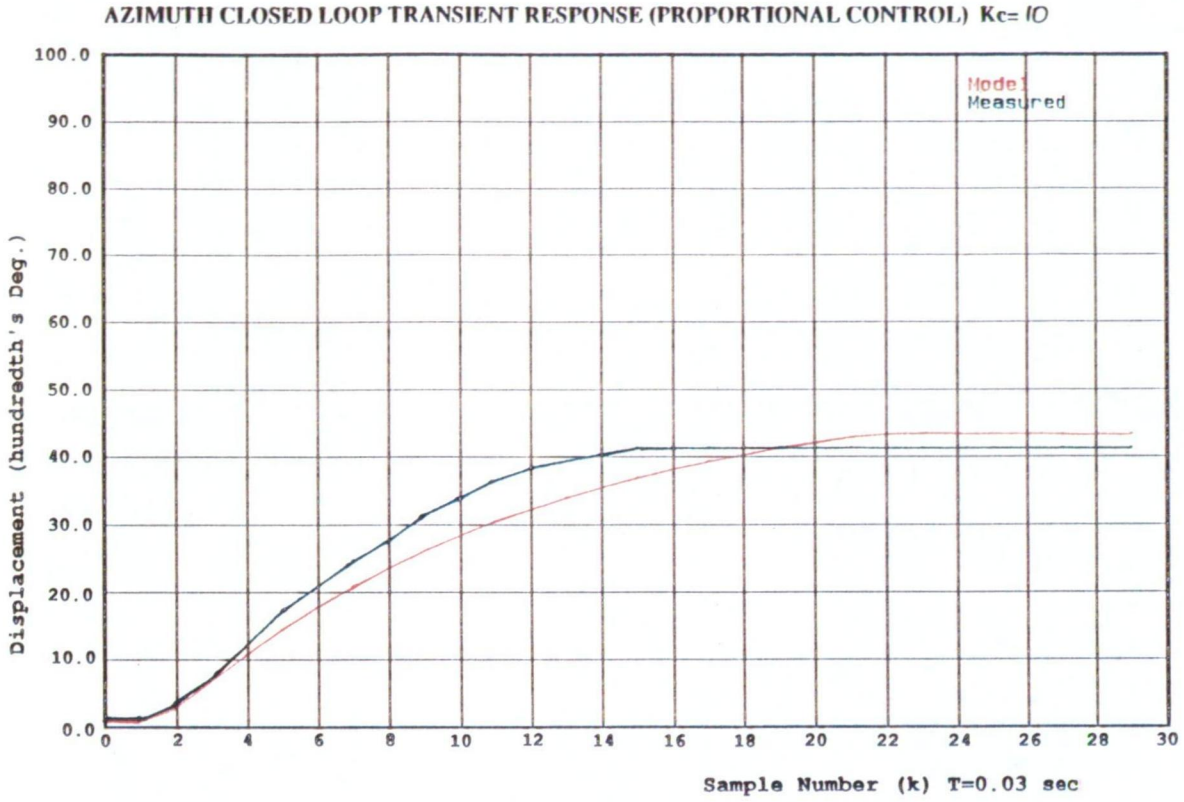
Graph 4.3B: Closed loop transient step response of the model (red) and the measured system (green) using proportional control at gain Kc=30. The measurement model is included in the feedback. See fig.4.5.

Graph 4.3C: Closed loop transient step response of the model (red) and the measured system (green) using proportional control at gain Kc=40. The measurement model is included in the feedback. See fig.4.5.

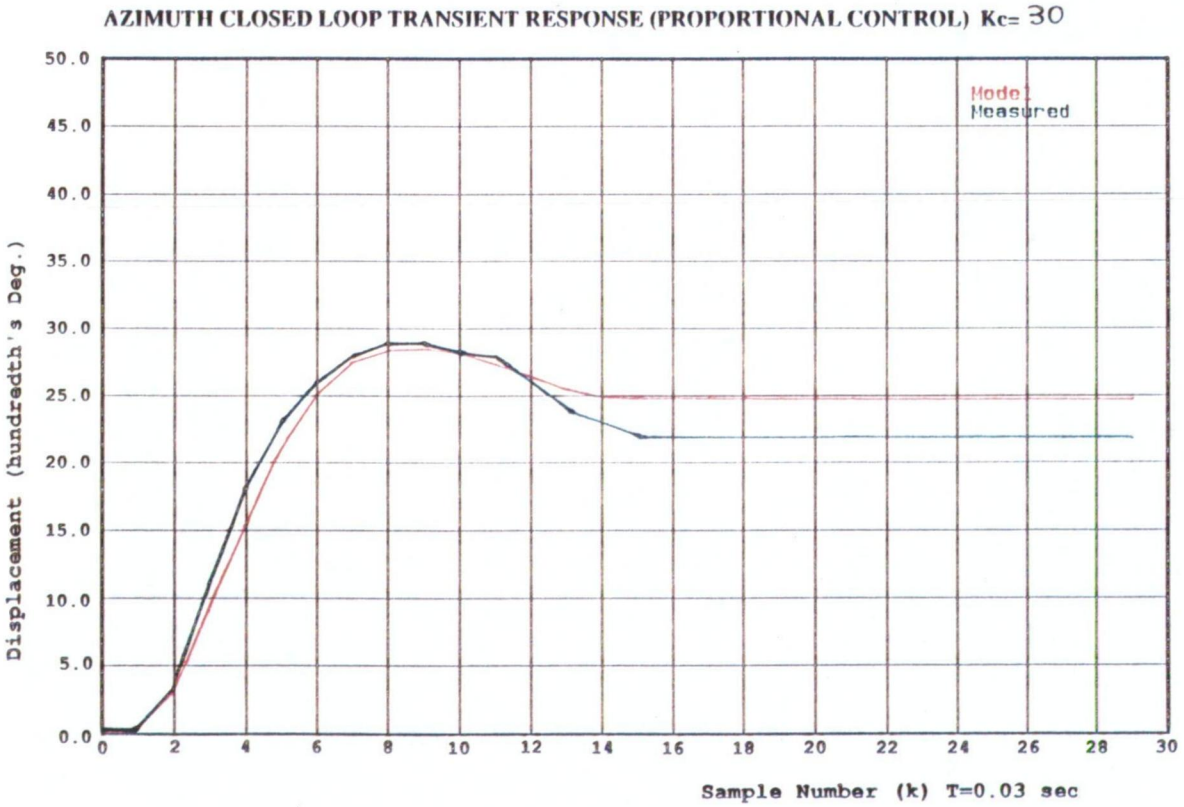
Graph 4.3D: Target acquisition simulation using proportional control with gain Kc=50, the red curve is the tracking system with infinite field of view, the blue curve is the tracking system with R=0.5 degrees. The black curve is the reference (position of the target).

Graph 4.3E: Target acquisition simulation using proportional control with gain Kc=100, the red curve is the tracking system with infinite field of view, the blue curve is the tracking system with R=0.5 degrees. The black curve is the reference (position of the target).

Graph.4.3A: step response of the closed loop system with the controller gain set to Kc=10

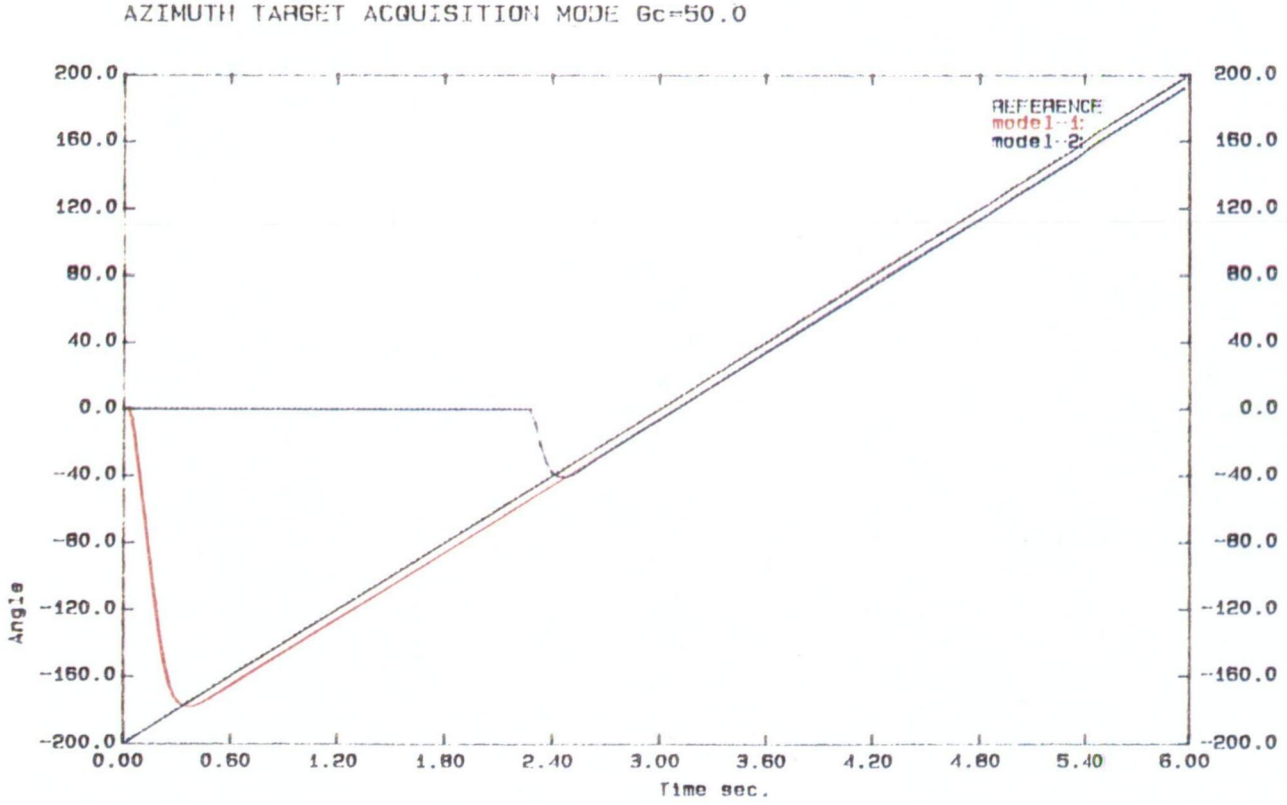


Graph.4.3B: step response of the closed loop system with the controller gain set to Kc=30

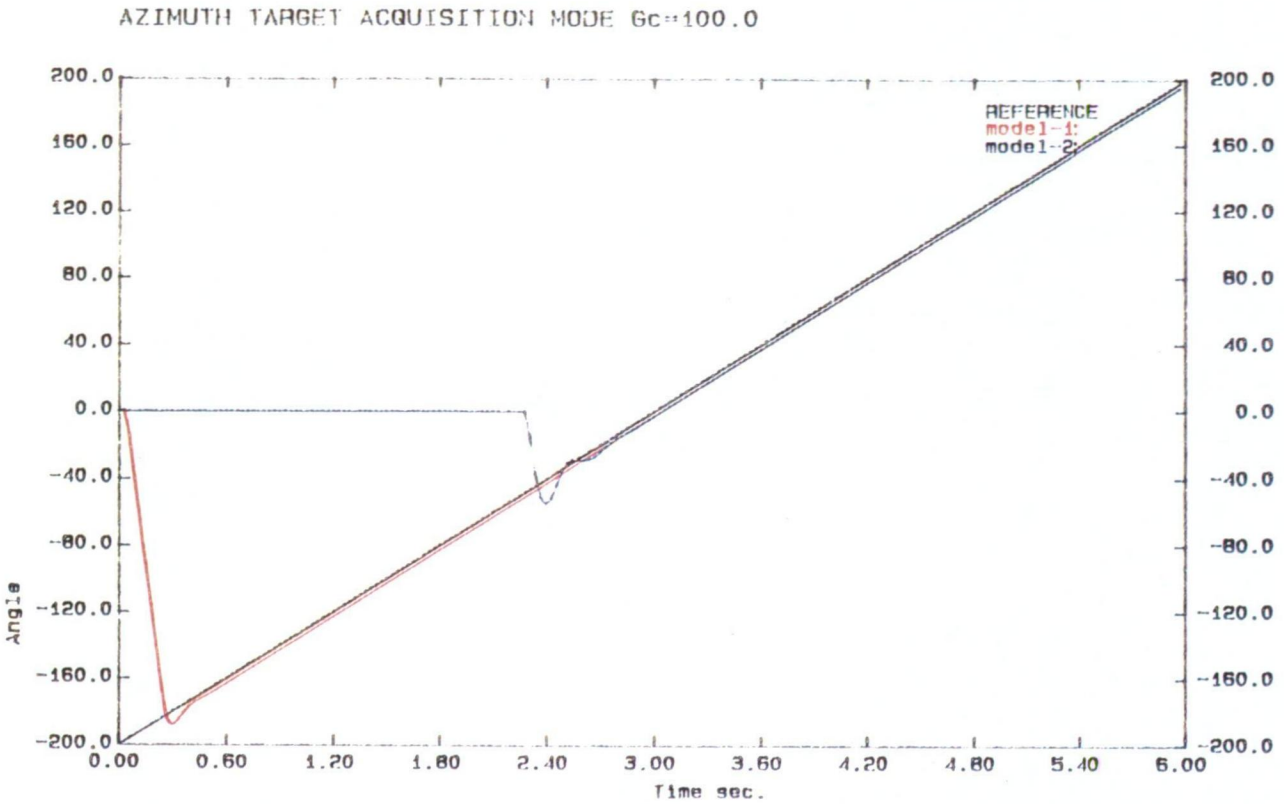


Graph.4.3C: step response of the closed loop system with the controller gain set to Kc=40

Graph.4.3D: Azimuth target acquisition response with controller gain Kc=50



Graph.4.3E: Azimuth target acquisition response with controller gain Kc=100



[4.5] Pole Placement Controller:

(a) Introduction:

The pole placement design strategy attempts to find the polynomial coefficients of the controller such that the closed loop poles and zeros of the system are located according to some design specifications. A number of papers have being published, referer to 4.1, 4.2, 4.3 in section 7.2A of the appendix. The pole placement design procedure requires the specification of the ideal (desired) response: Gm(q)=Am(q)/Bm(q). Fig.4.13 below shows the structure of the pole-placement controller which uses a pre-compensator in the forward path and feedback compensator in the feedback path.

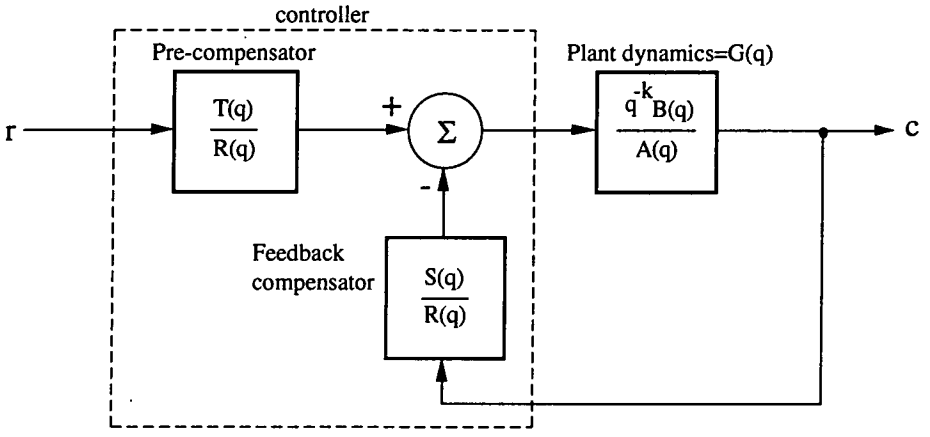


Fig.4.13 Block Diagram of the Pole Placement Controller

Where q^{-1} represents the backward shift (unit delay) operator. The advantage of this design strategy is that it enables the choice in selecting placement of both poles and zeros. Additionally non-minimum phase zeros can be taken into consideration so that pole/zero cancellation does not occour, similarily zeros which are not well defined or underdamped are not cancelled. The term r and c in fig.4.13 represent the input reference and the output response (either azimuth or elevation) of the system. The following delay operator notation for descrete time systems is used for all polynomials:

$$A(q) = 1 + a_1.q^{-1} + a_2.q^{-2} + ... a_n.q^{-n} \qquad (4.16)$$

Define G(q) be the open loop plant dynamics which is the product of the servosystem model H(z) and the measurement process M(z), thus G(z)=H(z).M(z) The transfer function is written as:

$$G(q) = \frac{q^{-k}.B(q)}{A(q)} \qquad (4.17)$$

Require the closed loop response Gm(q) to be: (where the user specifies Bm(q) and Am(q):

$$Gm(q) = \frac{q^{-k}.Bm(q)}{Am(q)} \tag{4.18}$$

From Fig.4.13 above, the closed loop transfer function is given by (the delay operator q is not shown for simplicity):

$$Gc = \frac{q^{-k}.T.B}{A.R + q^{-k}.S.B} \tag{4.19}$$

The advantages of this pole placement technique is that it enables both the placement of poles and zeros. Equating with the above expression, we obtain solutions to the R(q), T(q) and S(q) polynomials:

$$\frac{q^{-k}.Bm}{Am} = \frac{q^{-k}.T.B}{A.R + q^{-k}.S.B} \tag{4.20}$$

For non-minimum phase systems not all zeros can be cancelled, including zeros which are not well defined or underdamped. Therefore the B term must be written as the product of two separate terms. The first term is all the well defined zeros (which can be cancelled), the second term includes only zeros which cannot be cancelled by poles thus:

$$B = B^{+}.B^{-} \tag{4.21}$$

where: B^{+} = well defined (stable) zeros

B^{-} = zeros which cannot be cancelled

similarily, the desired controller term must include the same zeros which cannot be cancelled, thus we write this as the product of two terms, one contains identical non-minimum phase zeros:

$$Bm = Bm1 .B^{-} \tag{4.22}$$

substituting expressions 4.21 and 4.22 into equation 4.20 and cancelling the B^{+} term, the equation simplifies to:

$$\frac{q^{-k}.Bm1.B^{-}}{Am} = \frac{q^{-k}.T.B^{-}}{A.R1 + q^{-k}.S.B^{-}} \tag{4.23}$$

Equating gives solutions to the controller polynomials:

$$A.R1 + q^{-k}.S.B^{-} = Am \quad (4.24A)$$

$$T = Bm1 \quad (4.24B)$$

$$R = R1.B^{+} \quad (4.24C)$$

The polynomials R1(q) and S(q) must have the following degree such that the solution is causal with no extra delay:

$$deg(S) = deg(A) - 1 \quad (4.25A)$$

$$deg(R1) = deg(Am) + deg(Ao) - deg(A) \quad (4.25B)$$

It is assumed that polynomials A(q) and Am(q) are both monic, additionally A(q) and B(q) are assumed to be coprime, with Am(q) and Bm(q) also coprime.
The advantages of pole placement design controllers over self-tuning , minimum variance regulator controllers is robustness against time varying dead-time , and the ability to handle non-minimum phase system without modification. Additionally, the engineering specifications can be made to relate more easily to the poles assigned, whereas the tuning factors of the minimum variance controller are difficult to select.

(b) Simulation Results:

Plant Dynamics: The system G(z) which represents our model of the servosystem dynamics is described by the product: G(z)=H(z).M(z) (for the azimuth system) is given by:

$$G(z) = \frac{0.0035(z + 1.2)}{(z - 1)(z - 0.12)} \frac{0.607255\ (z + 0.253)}{z^2 - 0.429753\ z + 0.19308}$$

The first part of the equation describes the servomotor dynamics and the gain of the interface card, the second part is the measurement model developed in the previous section. Thus from equation 4.17:

$$k=2 \quad (4.27A)$$

$$B(z) = 0.0021254\ (1 + 1.2\ z^{-1})(1 + 0.253\ z^{-1}) \quad (4.27B)$$

$$A(z) = (1 - z^{-1})(1 - 0.12\ z^{-1})(1 - 0.429753\ z^{-1} + 0.19308\ z^{-2}) \quad (4.27C)$$

We do not wish to cancel the zero outside the unit circle, thus the B(z) term is written as a product of two terms, one contains non-minimum phase zero term which will not to be cancelled, the other contains stable (underdamped) zeros which may be cancelled:

$$B^{+} = (1 + 0.253\, z^{-1}) \tag{4.28A}$$

$$B^{-} = 0.0021254\,(1 + 1.2\, z^{-1}) \tag{4.28B}$$

-Model: Require closed loop poles to be located as given by fig.4.15, as a polynomial product term containing six poles located at values p1, p2, p3, p4, p5, p6 where complex conjugate pairs may also be included. Thus the model is described by the equation:

$$Gm(q) = \frac{Bm(q)}{Am(q)} \tag{4.29}$$

required closed loop response is:

$$Gm(z) = \frac{(0.0035\, z + 0.0042)\,(0.607255\, z + 0.153779)}{(z + p1)(z + p2)(z + p3)(z + p4)(z + p5)(z + p6)} \tag{4.30}$$

the poles can be placed anywhere, we chose four real poles and one pair of complex conjugate. Fig.4.15 below illustrates the mapping:

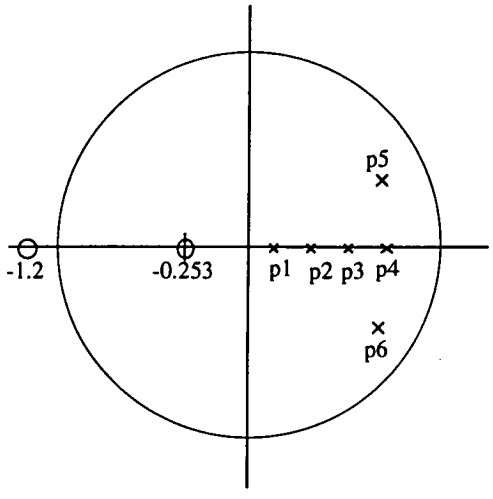


Fig.4.15 Typical Placement of Poles by Controller.

expanding the denominator pole terms into polynomial form:

$$Am(z) = am_6.z^6 + am_5.z^5 + am_4.z^4 + am_3.z^3 + am_2.z^2 + am_1.z^1 + am_0 \tag{4.31}$$

equating to the plant model solves for the R(q) and S(q) polynomials:

$$(a_4.z^4 + a_3.z^3 + a_2.z^2 + a_1.z^1 + a_0)(r_2.z^2 + r_1.z^1 + r_0) + z^2(s_3.z^3 + s_2.z^2 + s_1.z^1 + s_0)(b_1.z^1 + b_0) = Am(z)$$

we can equate each coefficient and obtain 7 simultaneous equations, terms for the R(q) and S(q) polynomials are solved by setting up a matrix giving the coefficients to the R(q) and S(q) polynomials. The T(q) polynomial is however a constant chosen such that the closed loop DC gain of Gm=1. A simulation program has been written: **SYSTPPD.C** to implement the above pole placement design scheme. The user has to simply select the placement of poles and zeros, the program automatically generates the coefficients to S(q), R(q) and T(q) polynomials. A copy of the simulation program is found in the appendix 7.4E

$$\begin{bmatrix} am_0 \\ am_1 \\ am_2 \\ am_3 \\ am_4 \\ am_5 \\ am_6 \end{bmatrix} = \begin{bmatrix} a_0 & 0 & 0 & 0 & 0 & 0 & 0 \\ a_1 & a_0 & 0 & 0 & 0 & 0 & 0 \\ a_2 & a_1 & a_0 & b_0 & 0 & 0 & 0 \\ a_3 & a_2 & a_1 & b_1 & b_0 & 0 & 0 \\ a_4 & a_3 & a_2 & 0 & b_1 & b_0 & 0 \\ 0 & a_4 & a_3 & 0 & 0 & b_1 & b_0 \\ 0 & 0 & a_4 & 0 & 0 & 0 & b_1 \end{bmatrix} \begin{bmatrix} r_0 \\ r_1 \\ r_2 \\ s_0 \\ s_1 \\ s_2 \\ s_3 \end{bmatrix}$$

The following graphs illustrate results obtained from the simulation by different placement of poles:

Simulation Results:

Graph 4.4A: BLUE:

Pole placement controller simulation with the poles placed at:

Poles: 0.5, 0.5, 0.5, 0.5, 0.70773+j0.21893, 0.70773-j0.21893

$S(z) = -746.3\ z^3 + 1103.2\ z^2 - 527.8\ z + 172.0$

$R(z) = 2.19351\ z^3 + 1.44010\ z^2 + 1.12378\ z + 0.22785$

GREEN:

Pole placement controller simulation with the poles placed at:

Poles: 0.2, 0.2, 0.2, 0.2, 0.70773+j0.21893, 0.70773-j0.21893

$S(z) = -57.64\ z^3 + 87.79\ z^2 - 25.14\ z + 2.09$

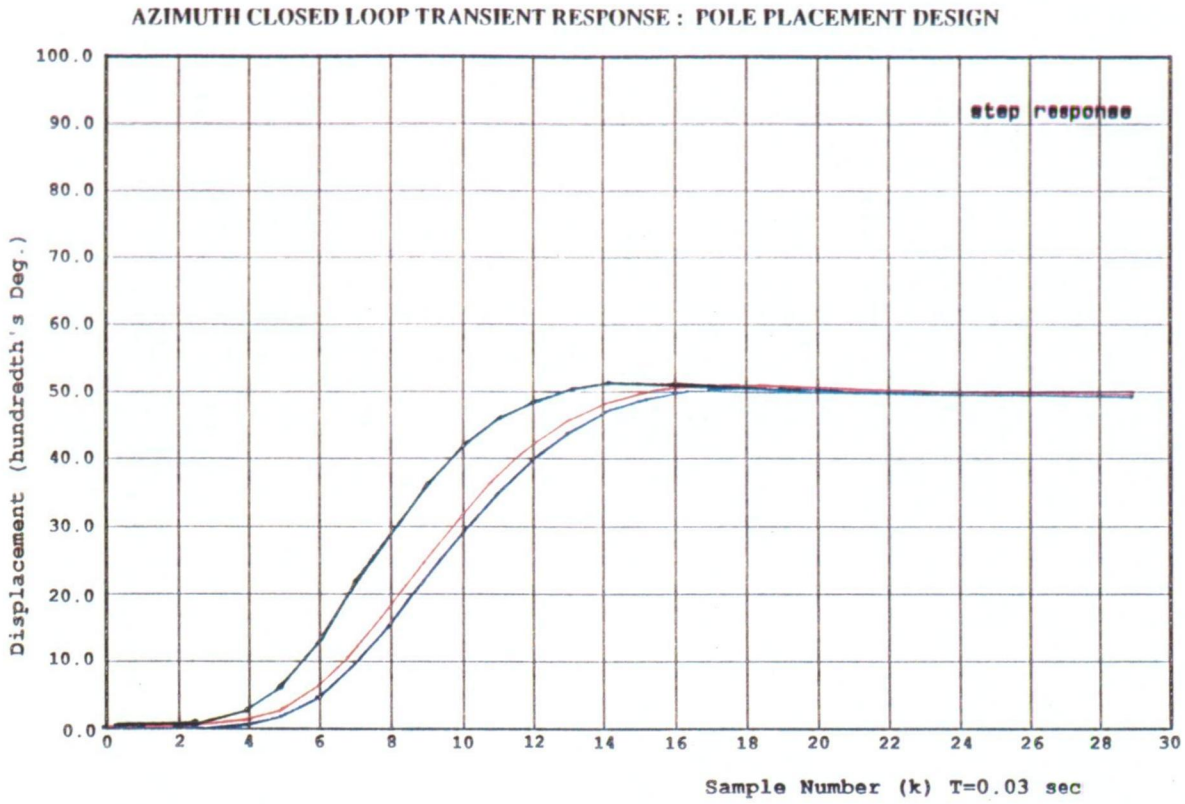
$R(z) = 0.72977\ z^3 - 0.06895\ z^2 - 0.04123\ z + 0.00583$

RED:
Pole placement controller simulation with the poles placed at:
Poles: 0.2, 0.3, 0.4, 0.5, 0.70773+j0.21893, 0.70773-j0.21893
$S(z) = -66.71\ z^3 + 143.85\ z^2 - 82.34\ z + 17.11$
$R(z) = 0.749054\ z^3 - 0.475448\ z^2 + 0.0044319\ z + 0.043748$

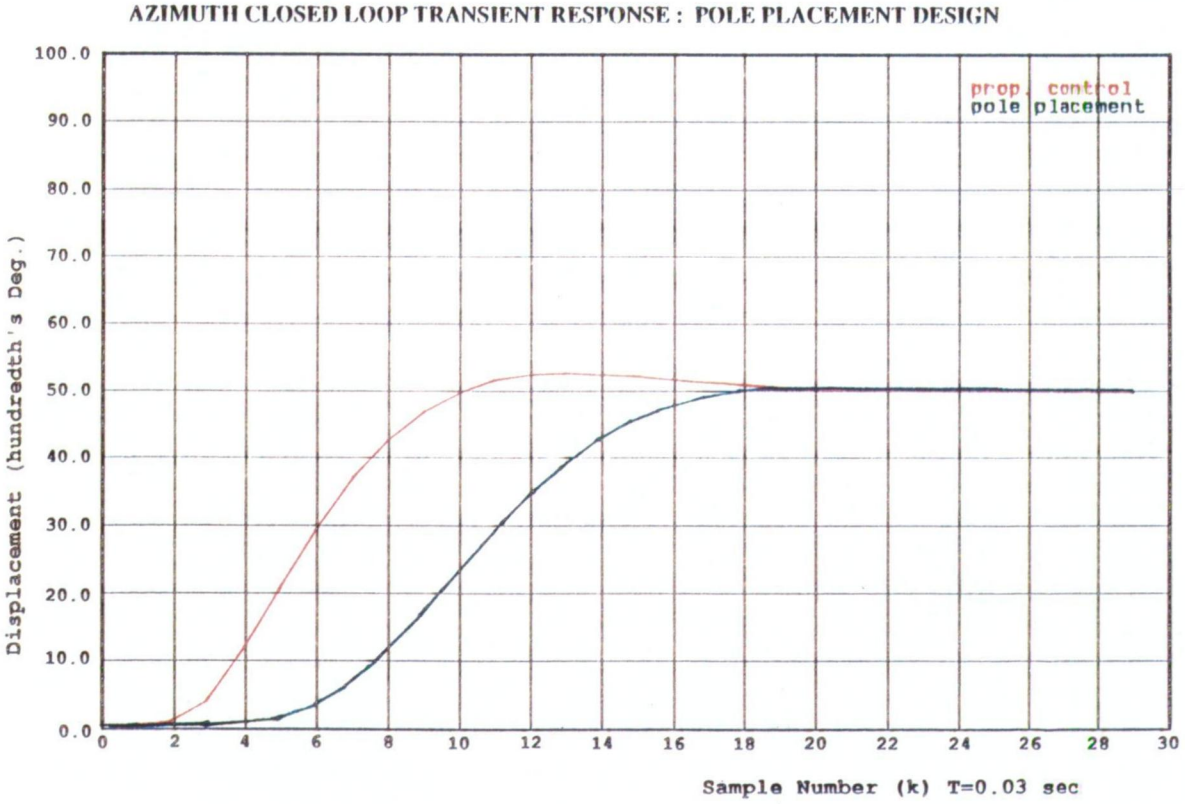
Graph 4.4B: Comparing the response from the pole placement controller and the proportional controller with the following values:
Pole placement controller simulation with the poles placed at:
Poles: 0.1, 0.1, 0.2, 0.2, 0.4+j0.35, 0.4-j0.35
Proportional controller gain Kc=20
The red curve is the pole placement controller step response, the green graph is the pole placement controller step response.

Graph 4.4C: Comparing the response from the pole placement controller and the proportional controller with the following values:
Pole placement controller simulation with the poles placed at:
Poles: 0.3, 0.4, 0.5, 0.6, 0.70773+j0.21893, 0.70773-j0.21893
Proportional controller gain Kc=20
The red curve is the pole placement controller step response, the green graph is the pole placement controller step response.

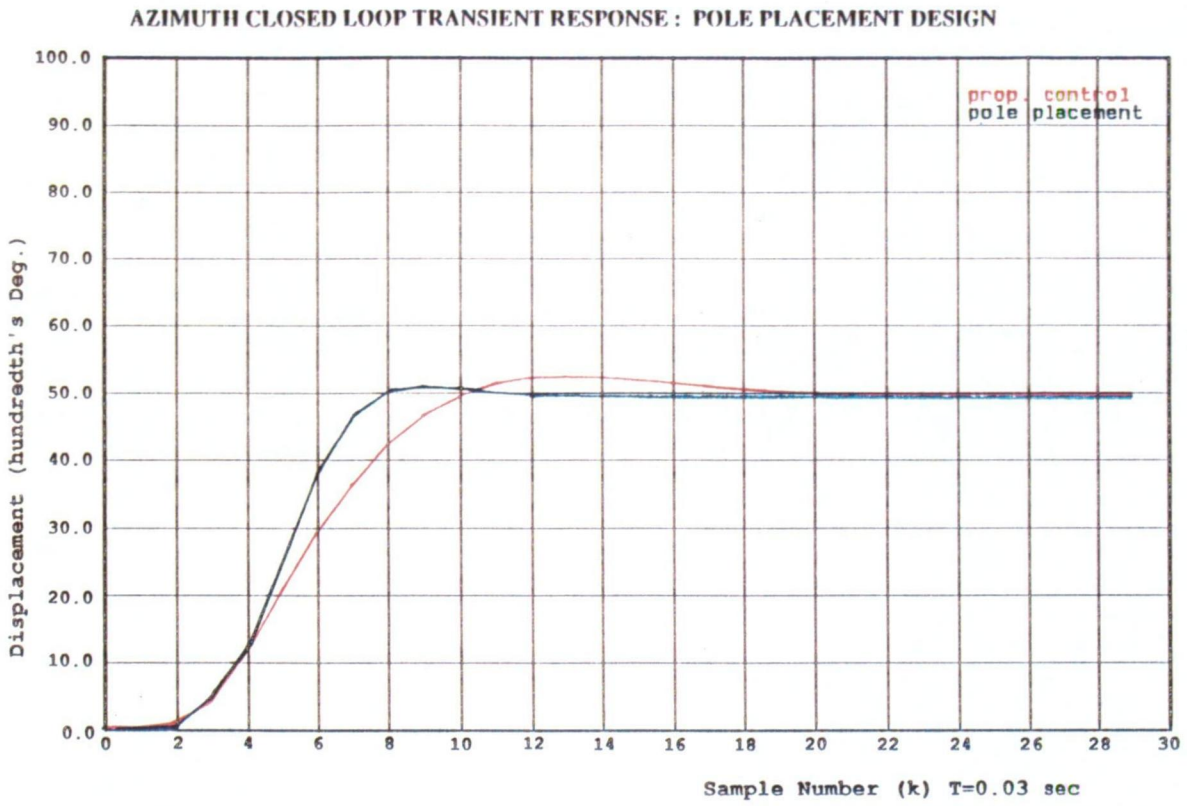
Graph 4.4A: Step response of the closed loop system with the pole-placement controller:



Graph 4.4B: Comparing proportional control (Kc=20) with the pole-placement controller:



Graph 4.4C: Comparing proportional control (Kc=20) with the pole-placement controller:



(c) Implementation on Anglescan Tracking System:

The pole placement controller has not yet being tested on the actual anglescan tracking system, only simulation results are available. Testing remains part of the future development work (refer to chapter 6). However, before the controller can be tested, it must be changed into a form which can be implemented by the anglescan tracking computer. This is shown in fig.4.17.

We change from the pole placement controller structure shown in fig.4.16 to the more conventional form shown in fig 4.17 where Gc(z) is the controller implementation by the anglescan computer. To change, simply compare the transfer functions of each system and solve for Gc(z).

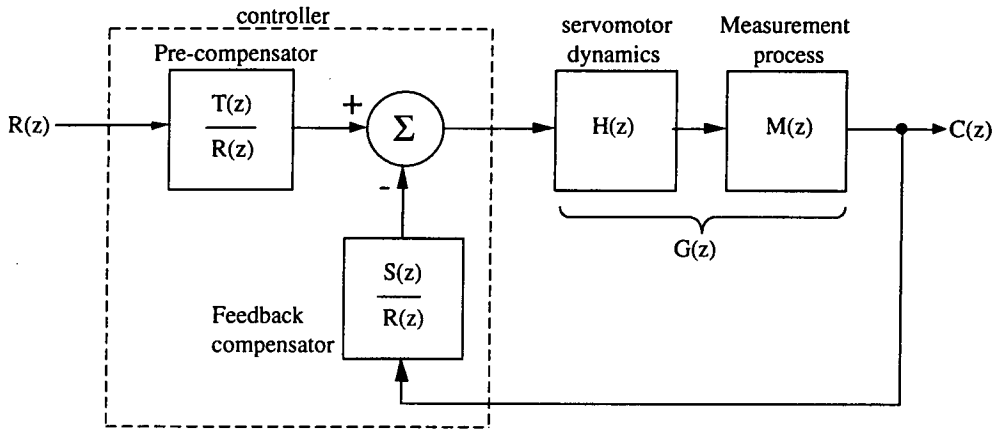


Fig.4.16 Block Diagram of the pole placement controller structure

Closed loop transfer function for the system shown in 4.16 is given by:

$$\frac{C(z)}{R(z)} = \frac{T(z).H(z).M(z)}{R(z) + S(z).H(z).M(z)} \qquad (4.32)$$

Where C(z) is the controller output displacement in radians (either azimuth or elevation), and R(z) is the input reference in radians.

The transfer function of the system shown in fig.4.17 is given by equation 4.33 below:

$$\frac{C(z)}{R(z)} = \frac{Gc(z).M(z).H(z)}{1 + Gc(z).M(z).H(z)} \qquad (4.33)$$

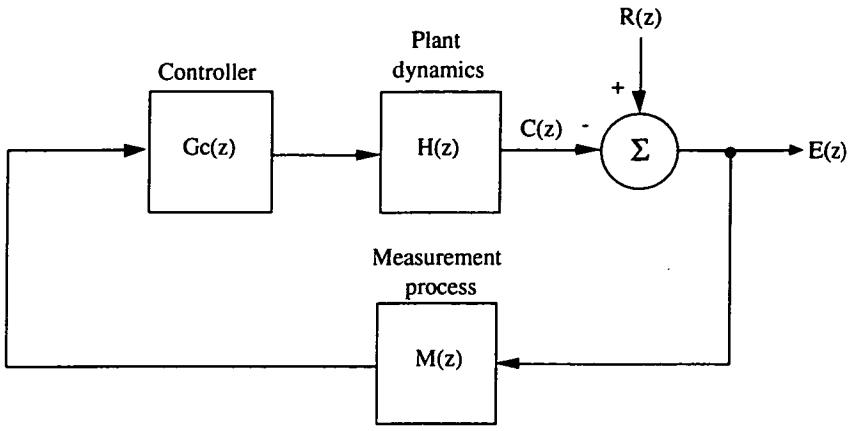


Fig.4.17 Pole placement controller structure -implementable form

Equating equations 4.32 and 4.33, we solve for Gc(z):

$$Gc(z) = \frac{T(z)}{S(z).M(z).H(z) + R(z) - T(z).M(z).H(z)} \qquad (4.34)$$

Which gives the structure of the controller polynomial Gc(z) which can be implemented on the anglescan tracking system. Substituting the polynomials for S(z), M(z) T(z) and R(z) found in section (c) into the above equation gives Gc(z) which is implemented by the anglescan computer tracking algorithms. Althought no experimental results are available for comparison, simulation model results have being provided if future work is to continue.

(d) Simulation Program:

A listing of the pole placement design and simulation program **SYSTPPD.C** is given in the appendix 7.4E. The program prompts the user to supply the plant model parameters and the desired placement of poles and zeros, it then computes the compensator polynomial coefficients and simulates the transient response of the model. The output is provided graphically to both the screen and HP plotter. The program consists of the following mathematical subroutines:

MultiplyPolyComplex():	Multiplies polynomials in the complex plane.
ExpandPoly():	Expands from factorized form to polynomial form.
SolveRSTPoly():	Solves for the R(z),S(z),T(z) polynomials
GraphInitLinear():	Initializes the graphics screen
GraphPlotLinear():	Plots the response on the screen
HPGLInitLinear():	Initializes the HP plotter
HPGLPlotLinear():	Plots the response on the HP plotter

[4.6] Chapter Summary:

The design of a controller is a twofold exercise involving the design and construction of the hardware, and the application of control theory to the analysis and design of an appropriate control algorithm. The design and simulation is based on previous results obtained from system identification tests (chapter-3).
Below we briefly summarize and conclude the historical development of the Anglescan tracking control system:

[1] The initial design consisted of a dedicated PID motion controller IC (LM628). This design is described in section 2.5 and a copy of the circuit diagram is provided in the appendix: sections 7.5E and 7.8K. The LM628 fully implements the PID compensator in hardware using 32 bit integer arithmetic. The microprocessor simply loads the 16 bit PID coefficients Kp, Ki, Kd into the LM628 which will then perform all the necessary closed loop feedback control functions. This method has the advantage of relieving the microprocessor from the task of implementing a controller in software, but restricts the design to that of a PID loop only. This method was tested and found to be unsatisfactory when tracking a moving target. Analysis using the TRIP simulation program revealed that the internal PID loop and the external (outer) software loop has the effect of placing the closed loop complex conjugate poles in the left hand inside the z plane making the control system unstable even at low controller gains.

[2] The design was subsequently changed to an open loop Pulse Width Modulator (PWM) as illustrated in figure 2.7 (chapter 2), the circuit diagram is shown in the appendix 7.5C. Since the design is open loop, the microprocessor can be made to implement a wide range of compensators.
This design has one added advantage, it includes the fine measurement logic which is used to obtain high resolution measurements (described in chapter 2.5). System identification was again applied to this servosystem.

[3] Using the above setup, we tested a range of controllers, the first is the proportional controller, with differing gains Kc. The response from the model was compared to the actual measured response and found to agree well, this is illustrated in section 4.4. In addition to proportional control, we also implemented an adaptive deadband compensation algorithm, a paper by C.Canudas (ref.4.8) describes this technique of adaptive deadband compensation in DC motor drives. Another paper by S.Yang (ref.4.9) describes a simpler method of adaptive pulse width control to overcome stiction and coulomb friction.

This scheme is based on the relationship between the displacement of a controlled output due to a single pulse input and the applied pulse width, this relationship is assumed to be linear. We tested this method on the anglescan system and found to actually overcome stiction, but was not practical for the tracking of a dynamically moving object. The method is only effective when the target is stationary.
There exists much literature on controller implemetations for automatic tracking systems. Most controllers use optimal linear quadratic control methods [refer to 4.21] for solar tracking systems, or optimal PID control [ref. 4.16]. Other papers including laser based automatic tracking systems [ref 4.7] describe optimal control solution by the Pontryagin maximum principle.

Similarily, much literature focuses on pole placement design techniques, for example complete pole-zero matching by K.J.Astrom and B.Whittenmark [ref. 4.1, 4.6, 4.5, 4.22, 4.3]. We investigated the pole placement design using the technique described by Astrom [ref. 4.3]. Simulation results show that the pole placement controller can exhibit better transient response compared with proportional control, but the desired poles must be carefully selected.

[4] The pole placement design was not tested on the actual anglescan system, however the simulation results are kept for future reference should this type of compensator be used.

Chapter-5

Instrument Calibration:

[5.1] Introduction:

Instrument calibration is an integral part of the anglescan software system as it serves to maintain the precision necessary for the proper operation of the instrument. Because the calibration technique applied is fairly complex, we have included a full description in this chapter. This chapter also extends some of the basic techniques previously developed from system identification theory (refer to chapter-3).

Instrument calibration attempts to determine as accurately as possible the following parameters which can only be initially estimated, refer to fig.5.2 for a pictorial description of the parameters listed below:

Field of View (R): The field of view determined by the amount of optical divergence produced by the rotating wedge prism, must be estimated to an accuracy equivalent to the instrument's positioning resolution, about 1 arc second. This value can be initially set to 0.5 degrees, ie the radius of the projected circle.

Index Pulse (Ip): The index pulse is generated by the rotary encoder mounted co-axially with the rotating wedge prism (refer to Fig.2.1). The rotary encoder produces 10000 counts per revolution, the index pulse must be estimated to about +/-1 count.

Centre of Field (Qx, Qy): The optical line of sight (centre of field) must be determined with respect to some arbitary (fixed) datum or reference, in both X and Y axis.

Horizontal Beam Error (θh): Since the horizontal (laser) beam cannot be assumed to be exactly horizontal (parallel to horizon), even when the instrument is fully levelled. This angle must be determined.

Vertical Beam Error (θv): Since the vertical (laser) beam cannot be assumed to be exactly vertical (parallel to zenith) even when the instrument is fully levelled. This angle must be determined.

The method used for calibration is similar to the method used for system identification ie: the application of least squares, but with several differences due to the nature of the equations in this case being non-linear and multivariable.

Thus our first step would be to linearize the equations by Taylor series expansion retaining only the first order (ie: linear) terms. All partial derivatives are subsequently grouped to form a Jacobian matrix. The unknowns are then solved for by iteration ie: by starting from some initial estimate and then repeatedly solving for the unknowns until the solutions converge to an estimate with the necessary accuracy. As will be seen by simulation, the equations converge very rapidally.

To show that the equations are derived correctly, a simulation program was implemented in C on an IBM-PC. A copy of the code is listed in appendix 7.4A, the program is called: SYSTCAL2.C. This program enables us to determine the precision, rate of convergence and stability of the equations before actually implementing the calibration program on the Anglescan computer system. A description of the program including some results will be given in section 5.4 and 5.5 of this chapter.

(b) Calibration Procedure:

Fig.5.1 below illustrates the setup required to calibrate the instrument:

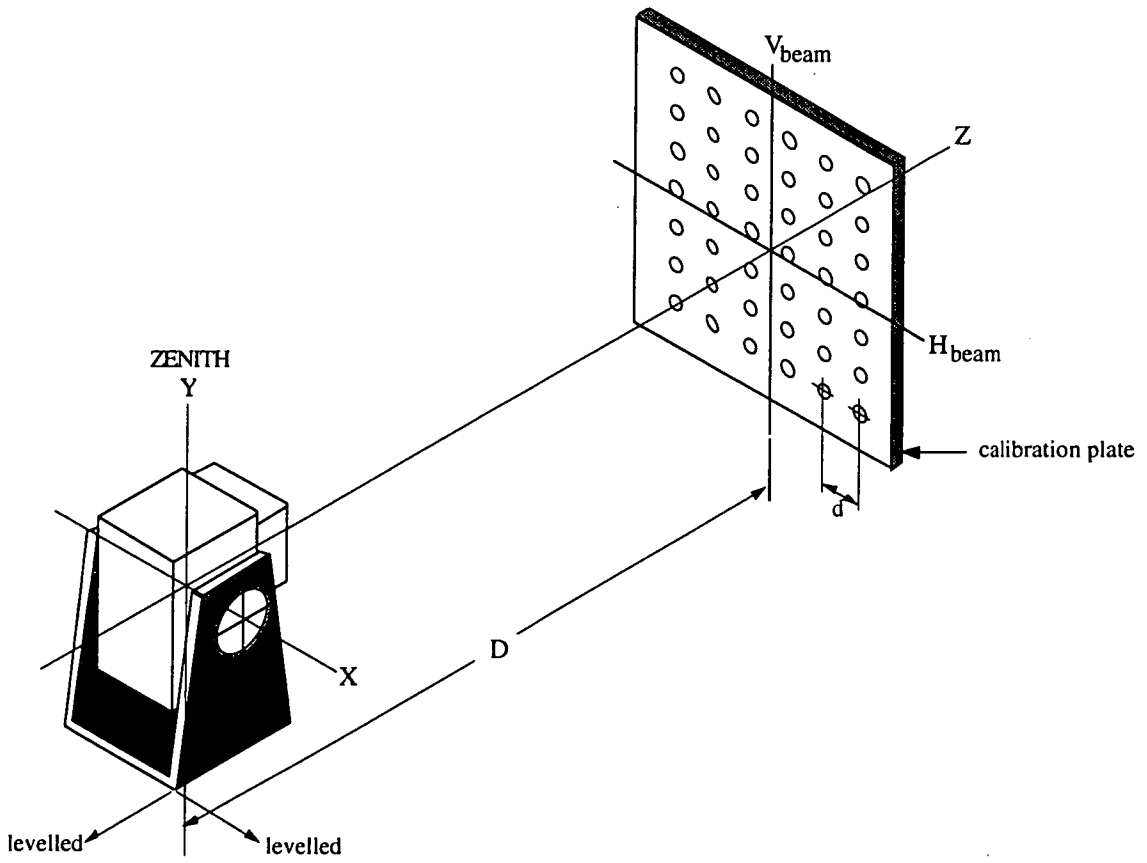


Fig.5.1 Anglescan Calibration Setup

We calibrate the instrument by using a specifically fabricated rectangular aluminium plate. The plate is machined in a precise rectangular grid of holes where the separation between hole centres is known. Both the Anglescan and calibration plate can be mounted on a standard tripod and adjusted with the three foot screws (refer to Fig.1.1 in chapter 1) until both anglescan and calibration plate are levelled. The calibration then proceeds by placing a retroreflective mirror in any one of the holes on the calibation plate and systematically taking measurements. This procedure is generally repeated a number of times (say for 10 different mirror locations). Increasing the number of data points measured will generally provide increased calibration accuracy.

Note that during the calibration procedure, the user must know accurately the angular separation between the holes on the calibration plate. The angular separation is calculated from knowing the distance between the anglescan and the calibration plate D, and separation between hole centres d.
All measurement points on the plate are referenced to an arbitary (fixed) datum, the instrument at the end of the calibration, will calculate the centre of field of view with respect to this datum.
Calibration software is initially written and tested in C on an IBM-PC, it will eventually be transfered to the anglescan on board computer, refer to program SYSTCAL2.C in the appendix 7.4A. The following notation and nomenclature will be used throughout this chapter:

x,y	Target (mirror) co-ordinates in angular units, either degrees or radians. This represents azimuth and elevation respectively.
$h_x(\underline{\theta})$	Set of equations relating the unknown calibration coefficients to the azimuth measurements.
$h_y(\underline{\theta})$	Set of equations relating the unknown calibration coefficients to the elevation measurements.
$\underline{\theta}$=[Qx, Qy, ϕv, ϕh, Ip, R]	Unknow vector, the coefficients are the unknowns to be determined from calibration.
$\underline{\alpha} = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$	Are the measurements obtained from the anglescan optical measurement system.

(c) Chapter Outline:

Derivation: Firstly, we derive the full set of calibration equations as a function of the unknowns described on page 2 of this chapter, and the target co-ordinates (known from measurements). In general we can write the azimuth and elevation (x,y) as some fumction h() of the unknown calibration parameters $\underline{\theta}$ and the measurement vector $\underline{\alpha}$:

$$x = h_x(\underline{\theta}, \underline{\alpha}) \quad (5.1A)$$

$$y = h_y(\underline{\theta}, \underline{\alpha}) \quad (5.1B)$$

where x and y represent the target co-ordinates in cartesian space, and the $\underline{\theta}$ represents the unknown vector such that: $\underline{\theta}=[Q_x, Q_y, \phi_v, \phi_h, I_p, R]$ and $\underline{\alpha} = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$
Next we linearize each of the equation by Taylor series expansion and retaining only the first order terms. All partial derivative terms are grouped to form a Jacobian matrix which we solve by iteration using least squares solution at each iteration. The solution is obtained when the difference between the previous value and the current value is below a certain error threshold.

Verification: The program is verified by simulation. To test the program, we first generate a number of simulated random target co-ordinates (placed within the field of view of the instrument). The program uses these co-ordinates to calculate the four measurements which the anglescan instrument is expected to read ie: $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$. These measurements are calculated from simple geometry (refer to fig.5.2). Note that the unknown vector $\underline{\theta}$ must be known in order to initially calculate the four measurements.
Finally, the calibration algorithm is applied to the four measurements and the unknown vector should converge to the value which was used to initially calculate the four input measurements from geometry.

[5.2] Derivation of Calibration Equations:

Fig.5.2 below illustrates the anglescan vertical and horizontal beams intersecting the target at the points: $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$. Both vertical and horizontal beams are tilted by ϕv and ϕh from the true vertical and true horizontal respectively. The index pulse is also shown to occur at an angle of Ip before the top dead centre. The other two unknowns are the location of centre of field with respect to some fixed arbitary datum (Qx,Qy), and the field of view R. All four measurements $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$ are referenced with respect to the position of the index pulse. The cross rotates clockwise along the circular track. The position of the centre of the cross each time one of the beams intersects the target occurs at 4 points marked P1, P2, P3, P4 corresponding to the four measurements.

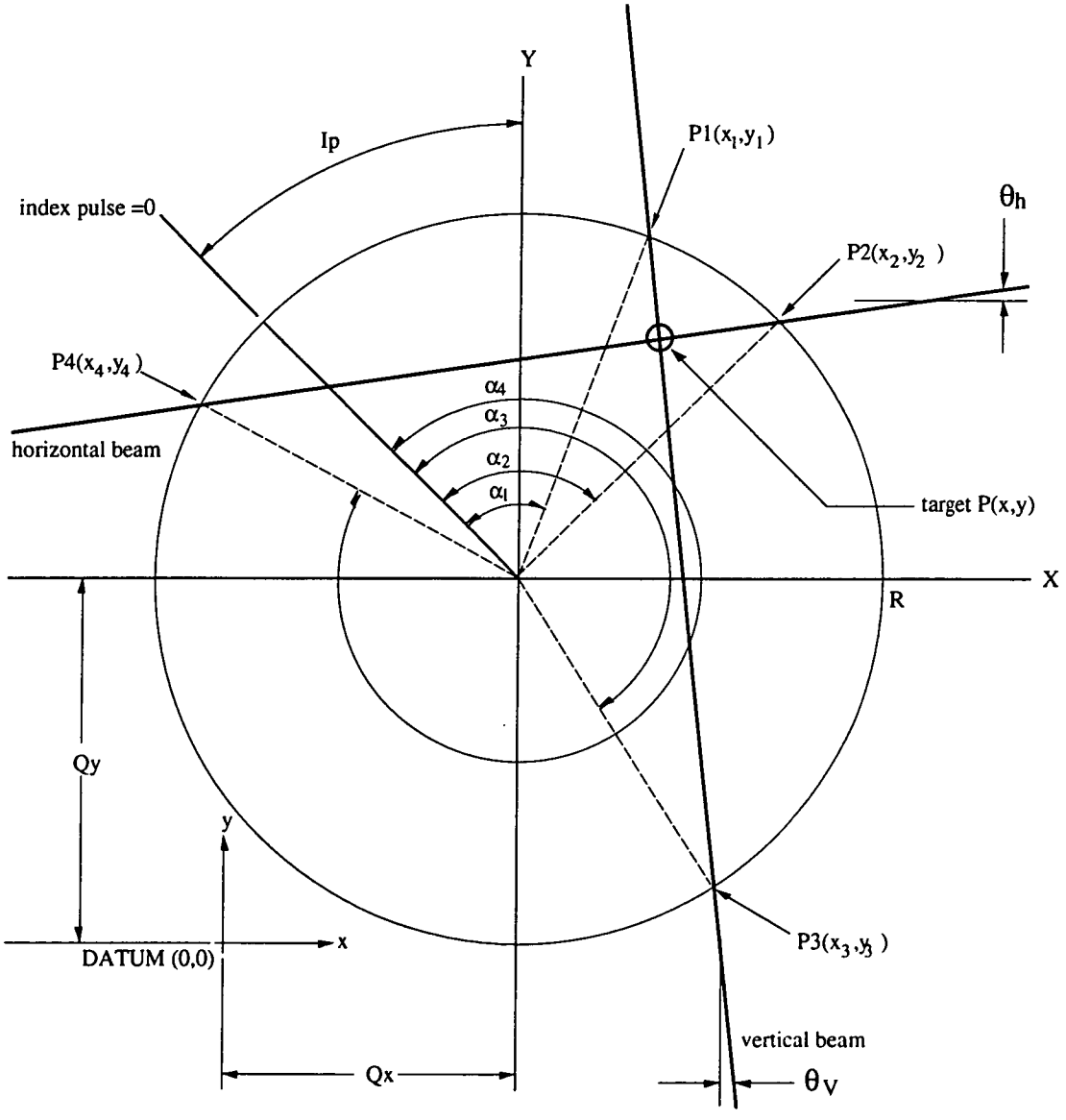


Fig.5.2 Calibration: Notation of Symbols.

The unknown vector $\underline{\theta}$ comprises of the six unknows described previously. The relationship between measurement vector $\underline{\alpha}$ and the known data points P(x,y) can be defined by the following functions (from equation 5.1A,B):

$$x = h_x(\underline{\theta}, \underline{\alpha}) + v \quad (5.2A)$$

$$y = h_y(\underline{\theta}, \underline{\alpha}) + v \quad (5.2B)$$

Where:

the unknown vector: $\underline{\theta} = \{Qx, Qy, \theta v, \theta h, Ip, R\}$

the measurement vector: $\underline{\alpha} = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$

and the v term represents the measurement error or residual. Both equations $h_x()$ and $h_y()$ are non linear. The components x,y are referenced to some arbitary datum as shown in Fig.5.2 above. The slope of the horizontal and vertical beams can be calculated from:

The slope of the horizontal beam is: $\tan(\theta h)$

The slope of the vertical beam is: $1/\tan(\theta v)$

The straight line equations for each beam can be described by the following functions for each measurement and each point:

$$(y - y_1) = \frac{1}{\tan(\theta_v)}(x - x_1) \quad (5.3A)$$

$$(y - y_3) = \frac{1}{\tan(\theta_v)}(x - x_3) \quad (5.3B)$$

$$(y - y_2) = \tan(\theta_h)(x - x_2) \quad (5.3C)$$

$$(y - y_4) = \tan(\theta_h)(x - x_4) \quad (5.3D)$$

Where each of the values of x_i and y_i can be calculated from the measured values of α_1, α_2, α_3, α_4. Thus for i=1,2,3,4 we have:

$$x_i = R.\sin(\alpha_i + I_P) \quad (5.4A)$$

$$y_i = R.\cos(\alpha_i + I_P) \quad (5.4B)$$

Note that rather than working with $\tan(\theta h)$ and $\tan(\theta v)$, because they represent slope, then if we redefine slope from equations 5.5A and B below to be:

$$\frac{1}{\phi_v} = \frac{\Delta Y}{\Delta X} = \frac{1}{\tan(\theta v)} \tag{5.5A}$$

$$\phi_h = \frac{\Delta Y}{\Delta X} = \tan(\theta h) \tag{5.5B}$$

Thus, rather than representing slope in degrees, the two: ϕh and ϕv are now representing the (dimensionless) slope of the horizontal and vertical beams respectively. Solving the two set of equations 5.3A and 5.3B simultaneously, then doing the same for equations 5.3C and 5.3D produces target co-ordinates as a function of the unkowns to be estimated, thus:

For the points $P1(x_1,y_1)$ and $P2(x_2,y_2)$ we get the equations:

$$x = \frac{\phi_v}{1 - \phi_v.\phi_h}(y_2 - y_1) - \frac{\phi_v.\phi_h}{1 - \phi_v.\phi_h}(x_2) + \frac{1}{1 - \phi_v.\phi_h}(x_1) \tag{5.6A}$$

$$y = \frac{\phi_h}{1 - \phi_v.\phi_h}(x_1 - x_2) - \frac{\phi_v.\phi_h}{1 - \phi_v.\phi_h}(y_1) + \frac{1}{1 - \phi_v.\phi_h}(y_2) \tag{5.6B}$$

the same applies to points $P3(x_3,y_3)$ and $P4(x_4,y_4)$, giving a total of four equations:

$$x = \frac{\phi_v}{1 - \phi_v.\phi_h}(y_4 - y_3) - \frac{\phi_v.\phi_h}{1 - \phi_v.\phi_h}(x_4) + \frac{1}{1 - \phi_v.\phi_h}(x_3) \tag{5.6C}$$

$$y = \frac{\phi_h}{1 - \phi_v.\phi_h}(x_3 - x_4) - \frac{\phi_v.\phi_h}{1 - \phi_v.\phi_h}(y_3) + \frac{1}{1 - \phi_v.\phi_h}(y_4) \tag{5.6D}$$

Inserting equations 5.3A and 5.3B into the above expressions results in a set of four equations relating target co-ordinates to measurements $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$ and to the unkown vector: {Qx, Qy, ϕv, ϕh, Ip, R}, and including arbitary datum reference terms:

Summary:

Below, the equations 5.7A,B,C,D represent the calibration functions:

$$hx = \frac{R.\phi_V}{1 - \phi_V.\phi_h}(\cos(\alpha_2+Ip) - \cos(\alpha_1+Ip)) - \frac{R.\phi_V.\phi_h}{1 - \phi_V.\phi_h}\sin(\alpha_2+Ip) + \frac{R}{1 - \phi_V.\phi_h}\sin(\alpha_1+Ip) + Qx \tag{5.7A}$$

$$hy = \frac{R.\phi_h}{1 - \phi_V.\phi_h}(\sin(\alpha_1+Ip) - \sin(\alpha_2+Ip)) - \frac{R.\phi_V.\phi_h}{1 - \phi_V.\phi_h}\cos(\alpha_1+Ip) + \frac{R}{1 - \phi_V.\phi_h}\cos(\alpha_2+Ip) + Qy \tag{5.7B}$$

$$h\bar{x} = \frac{R.\phi_V}{1 - \phi_V.\phi_h}(\cos(\alpha_4+Ip)-\cos(\alpha_3+Ip)) - \frac{R.\phi_V.\phi_h}{1 - \phi_V.\phi_h}\sin(\alpha_4+Ip) + \frac{R}{1 - \phi_V.\phi_h}\sin(\alpha_3+Ip) + Qx \qquad (5.7C)$$

$$h\bar{y} = \frac{R.\phi_h}{1 - \phi_V.\phi_h}(\sin(\alpha_3+Ip) -\sin(\alpha_4+Ip)) - \frac{R.\phi_V.\phi_h}{1 - \phi_V.\phi_h}\cos(\alpha_3+Ip) + \frac{R}{1 - \phi_V.\phi_h}\cos(\alpha_4+Ip)+ Qy \qquad (5.7D)$$

Equations 5.7A and 5.7B give target coordinates P(x,y) in terms of the two first measurements α_1, α_3. The second set of equations 5.7C, 5.7D give target coordinates in terms of the next set of measurements α_2, α_4. This implies that the equations maybe redundant, but from simulation results we will show that all the equations are required. This is due to the beams intersecting the target mirror form above and the left side for equations 5.7A,B and then from below and the right side of the mirror for equations 5.7C,D. If the target mirror diameter is small (less than 1 sec of arc) then the equations should be equal. If the diameter is large compared to R then the equations will not equal.

[5.3] Linearization and Solution by Least Squares:

In order to solve the unkown vector $\underline{\theta} = \{Qx, Qy, \phi v, \phi h, Ip, R\}$, the equations must first be linearized by finding the partial derivatives with respect to each unkown and then solving by ordinary least squares.
Linearization is by Taylor series expansion retaining only first order terms: applying to equations 5.1A and 5.1B we obtain:

$$x = h_x(\underline{\theta}) + \frac{\partial h_x(\underline{\theta}, \underline{\alpha})}{\partial \underline{\theta}}\Delta\underline{\theta} \qquad (5.8A)$$

$$y = h_y(\theta) + \frac{\partial h_y(\underline{\theta}, \underline{\alpha})}{\partial \underline{\theta}}\Delta\underline{\theta} \qquad (5.8B)$$

where the partial derivative term is evaluated at $\underline{\theta}_k$, note that this is a vector and must have partial derivatives with respect with each element (unkown). Thus: Equation 5.9A and 5.9B below give the full expression:

$$x = hx(\underline{\theta})+\frac{\partial hx(\underline{\theta})}{\partial Qx}\Delta Qx+\frac{\partial hx(\underline{\theta})}{\partial Qy}\Delta Qy+\frac{\partial hx(\underline{\theta})}{\partial \phi v}\Delta\phi v+\frac{\partial hx(\underline{\theta})}{\partial \phi h}\Delta\phi h+\frac{\partial hx(\underline{\theta})}{\partial Ip}\Delta Ip+\frac{\partial hx(\underline{\theta})}{\partial R}\Delta R \qquad (5.9A)$$

$$y = hy(\underline{\theta})+\frac{\partial hy(\underline{\theta})}{\partial Qx}\Delta Qx+\frac{\partial hy(\underline{\theta})}{\partial Qy}\Delta Qy+\frac{\partial hy(\underline{\theta})}{\partial \phi v}\Delta\phi v+\frac{\partial hy(\underline{\theta})}{\partial \phi h}\Delta\phi h+\frac{\partial hy(\underline{\theta})}{\partial Ip}\Delta Ip+\frac{\partial hy(\underline{\theta})}{\partial R}\Delta R \qquad (5.9B)$$

Where the Δ terms indicate the differences between the previous value of the estimate and the current value, for example: $\Delta R=(R_k - R_{k-1})$

The partial derivatives are also determined for the equations 5.7 C,D above.in the same manner.

(a) Partial derivatives:

The equations below represent partial derivatives with respect to all the unknown terms, there are a total of 6 unkowns and thus 6 partial derivatives:

For the equation of hx($\underline{\theta}$):

$$\frac{\partial hx}{\partial R} = \frac{\phi_v}{1 - \phi_v.\phi_h}(y_2 - y_1) - \frac{\phi_v.\phi_h}{1 - \phi_v.\phi_h}(x_2) - \frac{1}{1 - \phi_v.\phi_h}(x_1) \tag{5.10A}$$

$$\frac{\partial hx}{\partial \phi v} = \frac{R}{(1 - \phi_v.\phi_h)^2}(y_2 - y_1) - \frac{R.\phi_h}{(1 - \phi_v.\phi_h)^2}(x_2) - \frac{R.\phi_h}{(1 - \phi_v.\phi_h)^2}(x_1) \tag{5.10B}$$

$$\frac{\partial hx}{\partial \phi h} = \frac{R.\phi_v^2}{(1 - \phi_v.\phi_h)^2}(y_2 - y_1) - \frac{R.\phi_v}{(1 - \phi_v.\phi_h)^2}(x_2) - \frac{R.\phi_v}{(1 - \phi_v.\phi_h)^2}(x_1) \tag{5.10C}$$

$$\frac{\partial hx}{\partial Ip} = \frac{R.\phi_v}{1 - \phi_v.\phi_h}(-x_2 + x_1) - \frac{R.\phi_v.\phi_h}{1 - \phi_v.\phi_h}(y_2) + \frac{R}{1 - \phi_v.\phi_h}(y_1) \tag{5.10D}$$

$$\frac{\partial hx}{\partial Qx} = 1.0 \tag{5.10E}$$

$$\frac{\partial hx}{\partial Qy} = 0.0 \tag{5.10F}$$

the same applies to the set of equations describing hy($\underline{\theta}$) thus:

$$\frac{\partial hy}{\partial R} = \frac{\phi_h}{1 - \phi_v.\phi_h}(x_1 - x_2) - \frac{\phi_v.\phi_h}{1 - \phi_v.\phi_h}(y_1) - \frac{1}{1 - \phi_v.\phi_h}(y_2) \tag{5.11A}$$

$$\frac{\partial hy}{\partial \phi v} = \frac{R.\phi_h^2}{(1 - \phi_v.\phi_h)^2}(x_1 - x_2) - \frac{R.\phi_h}{(1 - \phi_v.\phi_h)^2}(y_1) + \frac{R.\phi_h}{(1 - \phi_v.\phi_h)^2}(y_2) \tag{5.11B}$$

$$\frac{\partial hy}{\partial \phi h} = \frac{R}{(1 - \phi_v.\phi_h)^2}(x_1 - x_2) - \frac{R.\phi_v}{(1 - \phi_v.\phi_h)^2}(y_1) - \frac{R.\phi_v}{(1 - \phi_v.\phi_h)^2}(y_2) \tag{5.11C}$$

$$\frac{\partial hy}{\partial Ip} = \frac{R.\phi_h}{1 - \phi_v.\phi_h}(y_1 + y_2) - \frac{R.\phi_v.\phi_h}{1 - \phi_v.\phi_h}(-x_1) + \frac{R}{1 - \phi_v.\phi_h}(-x_2) \tag{5.11D}$$

$$\frac{\partial hy}{\partial Qx} = 0.0 \tag{5.11E}$$

$$\frac{\partial hy}{\partial Qy} = 1.0 \tag{5.11E}$$

We can now set up the Jacobian matrices as following: for the linearized hx() function given by equation 5.9A, the Jacobian becomes **Hx**:

$$\mathbf{H_X} = \begin{bmatrix} \frac{\partial hx}{\partial R} & \frac{\partial hx}{\partial \phi v} & \frac{\partial hx}{\partial \phi h} & \frac{\partial hx}{\partial Ip} & \frac{\partial hx}{\partial Qx} & \frac{\partial hx}{\partial Qy} \\ & & & \vdots & & \\ \frac{\partial hx}{\partial R} & \frac{\partial hx}{\partial \phi v} & \frac{\partial hx}{\partial \phi h} & \frac{\partial hx}{\partial Ip} & \frac{\partial hx}{\partial Qx} & \frac{\partial hx}{\partial Qy} \end{bmatrix} \qquad (5.12A)$$

For each row, it indicates a set of measurements, similarily for the linearized hy() equation 5.9B, we set up a Jacobian **Hy** matrix:

$$\mathbf{H_Y} = \begin{bmatrix} \frac{\partial hy}{\partial R} & \frac{\partial hy}{\partial \phi v} & \frac{\partial hy}{\partial \phi h} & \frac{\partial hy}{\partial Ip} & \frac{\partial hy}{\partial Qx} & \frac{\partial hy}{\partial Qy} \\ & & & \vdots & & \\ \frac{\partial hy}{\partial R} & \frac{\partial hy}{\partial \phi v} & \frac{\partial hy}{\partial \phi h} & \frac{\partial hy}{\partial Ip} & \frac{\partial hy}{\partial Qx} & \frac{\partial hy}{\partial Qy} \end{bmatrix} \qquad (5.12B)$$

The first row is evaluated at the first measurement of the target: P(x,y), whereas the last row is evaluated at the last measurememt at P(x,y), given a set of N target measurements. Where P(x,y) represents the centre of the target with respect to some arbitary datum, refer back to fig.5.2.

(b) Solution By Least Squares:

Once the equations have being linearized, the standard methods developed in section 3.5 can now be applied directly. The only variation in this case is that we start with an initial estimate of the unknowns and solve recursively for a better estimate. All four equations (5.7A,B,C,D) must be considered and the sum square error simultaneously minimized in all the four equations 5.7A,B,C,D.

The iteration is repeated until the difference between the previous estimate and the current estimate is below a certain value. This test must be conducted for each unknown calibration coefficient.

General Linear Case:

Consider the case of a simple system of linear equations, the solution by ordinary least squares is found by writing the equations in the general form:

$$\underline{x} = \mathbf{A}.\underline{\theta} + \underline{e} \tag{5.13}$$

Where $\underline{x}$ is the observation, **A** matrix is the set of measurements, $\underline{\theta}$ is the unkown vector and $\underline{e}$ is the residual (error) vector. To minimize the sum of the errors squared using the cost function for n measurements, is the sum of the residuals for each k measurement:

$$J_n = \sum_{k=1}^{n} e_k^2 \tag{5.14}$$

which can be written as:

$$J_n = \sum_{k=1}^{n} (\underline{x} - \mathbf{A}.\underline{\theta})^2 \tag{5.15}$$

It can be shown that to minimize J_n with respect to $\underline{\theta}$, the solution has the form: (see appendix 7.3A for derivation).

$$\underline{\theta} = \left[\mathbf{A}^T.\mathbf{A} \right]^{-1}.\mathbf{A}.\underline{x} \tag{5.18}$$

Which is the general solution to the least squares problem.

Non Linear Case:

Now consider a system described by a set of non linear equations, where h() describes some non linear funtion with a continuous analytic partial derivative. This method is described in detail by Sorenson [ref. 5.2]. Again $\underline{x}$ is the observation vector, $\underline{e}$ is the residual, $\underline{\theta}$ is the unknown vector, and h() is some non linear function:

$$\underline{x} = h(\underline{\theta}) + \underline{e} \tag{5.18}$$

Applying linearization by Taylor series expansion and retaining only first order derivative terms evaluated at k'th iteration:

$$\underline{x} = h(\underline{\theta}_k) + \frac{\partial h(\underline{\theta}_k)}{\partial \underline{\theta}_k} \Delta \underline{\theta}_k \tag{5.19}$$

Where $\Delta\underline{\theta}_k = \underline{\theta}_k - \underline{\theta}_{k-1}$, redefine the partial derivative term as a Jacobian matrix thus:

$$\mathbf{H}_k = \frac{\partial h(\underline{\theta}_k)}{\partial \underline{\theta}_k} \tag{5.20}$$

then expression (5.19) can be re-written in the linear form:

$$\underline{x} = h(\underline{\theta}_k) + \mathbf{H}_k (\underline{\theta}_k - \underline{\theta}_{k-1}) \tag{5.21}$$

Equation 5.21 is now in the same form as equation 5.13, thus the solution to the above equation is similar to equation 5.18 and is therefore given by:

$$(\underline{\theta}_{k+1} - \underline{\theta}_k) = \left[\mathbf{H}_k^T . \mathbf{H}_k \right]^{-1} . \mathbf{H}_k . (\underline{x} - h(\underline{\theta}_k)) \tag{5.22}$$

Or in recursive format:

$$\underline{\theta}_{k+1} = \underline{\theta}_k + \left[\mathbf{H}_k^T . \mathbf{H}_k \right]^{-1} . \mathbf{H}_k . (\underline{x} - h(\underline{\theta}_k)) \tag{5.23}$$

Equation 5.23 recursively finds a better approximate to the unknown vector $\underline{\theta}_{k+1}$ from the previous value of $\underline{\theta}_k$ The procedure terminates when the term: $x-h(\underline{\theta})$ vanishes. Note that the number of unkowns is six, thus the matrix inversion in equation 5.11 is a 6x6 square matrix. We can also include a step factor $\lambda<1$ which determines the rate of convergence of equation 5.13 thus:

$$\underline{\theta}_{k+1} = \underline{\theta}_k + \left[\mathbf{H}_k^T . \mathbf{H}_k \right]^{-1} . \mathbf{H}_k . (\underline{x} - h(\underline{\theta}_k)) . \lambda \tag{5.24}$$

Multi Equations Case:

We next consider the case of multiple equations, as in our case there are a set of 4 equations. Consider the simpler case of only two equations as in 5.25A,B:

$$\underline{x} = hx(\underline{\theta}) + \underline{e}_x \tag{5.25A}$$

$$\underline{y} = hy(\underline{\theta}) + \underline{e}_y \tag{5.25B}$$

where the unknown vector $\underline{\theta}$ is the same in each equation, the error function is the sum of all the error terms combined which we want to minimize, thus the error function is:

$$J_n = \sum_{i=1}^{n}\left[(\underline{x} - hx(\underline{\theta}))^2 + (\underline{y} - hy(\underline{\theta}))^2 \right] \tag{5.26}$$

or:

$$J_n = \underline{e}_x^T \cdot \underline{e}_x + \underline{e}_y^T \cdot \underline{e}_y \tag{5.27}$$

Solution by least squares leads to the following equation:

$$\underline{\theta}_{k+1} = \underline{\theta}_k + \left[\mathbf{H}_x^T\mathbf{H}_x + \mathbf{H}_y^T\mathbf{H}_y \right]^{-1}\left[\mathbf{H}_x^T(\underline{x} - h_x(\underline{\theta})) + \mathbf{H}_y^T(\underline{y} - h_y(\underline{\theta})) \right] \tag{5.28}$$

(c) Summary of Equations:

Given the following set of equations which represent the position of the target P(x,y) as a function of the measurements $\underline{\alpha}$ and unknown vector $\underline{\theta}$:

$$hx = \frac{R.\phi_V}{1 - \phi_V.\phi_H}(\cos(\alpha_2+Ip)-\cos(\alpha_1+Ip)) - \frac{R.\phi_V.\phi_H}{1 - \phi_V.\phi_H}\sin(\alpha_2+Ip) + \frac{R}{1 - \phi_V.\phi_H}\sin(\alpha_1+Ip) \tag{5.29A}$$

$$hy = \frac{R.\phi_H}{1 - \phi_V.\phi_H}(\sin(\alpha_1+Ip)-\sin(\alpha_2+Ip)) - \frac{R.\phi_V.\phi_H}{1 - \phi_V.\phi_H}\cos(\alpha_1+Ip) + \frac{R}{1 - \phi_V.\phi_H}\cos(\alpha_2+Ip) \tag{5.29B}$$

$$h\bar{x} = \frac{R.\phi_V}{1 - \phi_V.\phi_H}(\cos(\alpha_4+Ip)-\cos(\alpha_3+Ip)) - \frac{R.\phi_V.\phi_H}{1 - \phi_V.\phi_H}\sin(\alpha_4+Ip) + \frac{R}{1 - \phi_V.\phi_H}\sin(\alpha_3+Ip) \tag{5.29C}$$

$$h\bar{y} = \frac{R.\phi_H}{1 - \phi_V.\phi_H}(\sin(\alpha_3+Ip)-\sin(\alpha_4+Ip)) - \frac{R.\phi_V.\phi_H}{1 - \phi_V.\phi_H}\cos(\alpha_3+Ip) + \frac{R}{1 - \phi_V.\phi_H}\cos(\alpha_4+Ip) \tag{5.29D}$$

We now write the jacobian matrices for the equations 5.29A,B,C,D where the partial derivatives are given by: 5.10A,B,C,D,E,F and 5.11A,B,C,D,E,F.respectively.

Jacobian for equation:5.29A:

$$\mathbf{H_x} = \begin{bmatrix} \frac{\partial hx}{\partial R} & \frac{\partial hx}{\partial \phi v} & \frac{\partial hx}{\partial \phi h} & \frac{\partial hx}{\partial Ip} & \frac{\partial hx}{\partial Qx} & \frac{\partial hx}{\partial Qy} \\ \vdots & & & & & \\ \frac{\partial hx}{\partial R} & \frac{\partial hx}{\partial \phi v} & \frac{\partial hx}{\partial \phi h} & \frac{\partial hx}{\partial Ip} & \frac{\partial hx}{\partial Qx} & \frac{\partial hx}{\partial Qy} \end{bmatrix} \tag{5.30A}$$

Jacobian for equation:5.29B:

$$\mathbf{Hy} = \begin{bmatrix} \frac{\partial hy}{\partial R} & \frac{\partial hy}{\partial \phi v} & \frac{\partial hy}{\partial \phi h} & \frac{\partial hy}{\partial Ip} & \frac{\partial hy}{\partial Qx} & \frac{\partial hy}{\partial Qy} \\ \vdots & & & & & \\ \frac{\partial hy}{\partial R} & \frac{\partial hy}{\partial \phi v} & \frac{\partial hy}{\partial \phi h} & \frac{\partial hy}{\partial Ip} & \frac{\partial hy}{\partial Qx} & \frac{\partial hy}{\partial Qy} \end{bmatrix} \tag{5.30B}$$

Jacobian for equation:5.29C:

$$\mathbf{H}_{\bar{x}} = \begin{bmatrix} \frac{\partial h\bar{x}}{\partial R} & \frac{\partial h\bar{x}}{\partial \phi v} & \frac{\partial h\bar{x}}{\partial \phi h} & \frac{\partial h\bar{x}}{\partial Ip} & \frac{\partial h\bar{x}}{\partial Qx} & \frac{\partial h\bar{x}}{\partial Qy} \\ \vdots & & & & & \\ \frac{\partial h\bar{x}}{\partial R} & \frac{\partial h\bar{x}}{\partial \phi v} & \frac{\partial h\bar{x}}{\partial \phi h} & \frac{\partial h\bar{x}}{\partial Ip} & \frac{\partial h\bar{x}}{\partial Qx} & \frac{\partial h\bar{x}}{\partial Qy} \end{bmatrix} \tag{5.30C}$$

Jacobian for equation:5.29C:

$$\mathbf{H}_{\bar{y}} = \begin{bmatrix} \frac{\partial h\bar{y}}{\partial R} & \frac{\partial h\bar{y}}{\partial \phi v} & \frac{\partial h\bar{y}}{\partial \phi h} & \frac{\partial h\bar{y}}{\partial Ip} & \frac{\partial h\bar{y}}{\partial Qx} & \frac{\partial h\bar{y}}{\partial Qy} \\ \vdots & & & & & \\ \frac{\partial h\bar{y}}{\partial R} & \frac{\partial h\bar{y}}{\partial \phi v} & \frac{\partial h\bar{y}}{\partial \phi h} & \frac{\partial h\bar{y}}{\partial Ip} & \frac{\partial h\bar{y}}{\partial Qx} & \frac{\partial h\bar{y}}{\partial Qy} \end{bmatrix} \tag{5.30D}$$

Solving by least squares gives the recursive function: Eqn.5.31 below

$$\underline{\theta}_{k+1} = \underline{\theta}_k + \mathbf{Ha}^{-1}.\mathbf{Hb} \tag{5.31}$$

Where the matrices are:

$$\mathbf{Ha} = \mathbf{Hx}^T.\mathbf{Hx} + \mathbf{Hy}^T.\mathbf{Hy} + \mathbf{H\bar{x}}^T.\mathbf{H\bar{x}} + \mathbf{Hy}^T.\mathbf{H\bar{y}}$$

$$\mathbf{Hb} = \mathbf{Hx}^T(x - hx) + \mathbf{Hy}^T(y - hy) + \mathbf{H\bar{x}}^T(\bar{x} - h\bar{x}) + \mathbf{H\bar{y}}^T(\bar{y} - h\bar{y})$$

[5.4] Computer Simulation:

(a) Description:

Simulation by computer will enable us to verify the calibration equations, determine the rate of convergence, accuracy and check the algorithm. The calibration program will eventually be part of the Anglescan software system. A copy of the simulation calibration program "SYSTCAL2.C" is listed in section 7.4A of the appendix, a description of the algorithm is illustrated on the following page (fig.5.3) in flowchart form. The program consists of the following sections (see fig 5.3 for blocks):

Block 1: Prompts the user to supply the six unkown calibration constants denoted by the values:

- R: Instrument's field of view in degrees.
- ϕv: Vertical beam error in slope units, not degrees.
- ϕh: Vertical beam error in slope units, not degrees.
- Ip: Index pulse {-180 to +180} degrees, can occour anywhere.
- Qx: Azimuth centre of field with respect to some known arbitary datum.
- Qy: Elevation centre of field with respect to some known arbitary datum.

Block 2: The computer generates 16 random target locations, each must be within the field of view of the instrument, the target locations are denoted by Pi(x,y) where i=1 to 16. This would in effect be the same as the user placing the target mirror at random points on the calibration plate and taking a reading with the anglescan.

Block 3: Using the 16 randomly generated target locations, and the previously supplied calibration constants, the program then computes the expected set of four measurements that the anglescan instrument would obtain for each target location.

This is calculated using geometry, thus for each target point P(x,y) ---> $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$. There are 16 sets of these measurements. Additionally we can introduce (random) noise to each measurement to enable the effect of noise to be determined.

Block 4: We input the 16 sets of $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$ measurements and apply the derived calibration equations and the iterative form of solution given by equation 5.31. This is the actual calibration algorithm.

The iteration terminates when either:
- -the term $|\underline{\theta}_{k+1} - \underline{\theta}_k|$ = SMALL.(user input)
- -the number of iterations > Nmax (user input)
- -the user generates a keyboard interrupt.

Note that in the first condition, since this is a vector, then all the elements of the vector are tested individually for this condition to hold true. Initially, pre-estimates are necessary to commence the iteration, pre-estimates are set to the following values:

R(0):	0.500	The field of view is fixed to 0.5 degrees by the optical system
ϕv(0):	0.000	The vertical beam error is assumed zero.
ϕh(0):	0.000	The horizontal beam error is assumed zero.
Ip(0):	Trial/error	Must be determined initially by trial / error.
Qx(0):	0.000	Assume that the x datum reference is the centre of the circle
Qy(0):	0.000	Assume that the y datum reference is the centre of the circle

The index pulse Ip initial value is found by a trial and error scheme. The index pulse can be anywhere in the range between -180 to +180 degrees equally likely. Its value depends on the mechanical construction and remains fixed. The program then applies the recursive calibration algorithms as described in section 5.3.

Block 5: Displays the results of the estimated unknown vector, comparing with the true value.

(b) Simulation Program:

A copy of the program is listed in the appendix, section 7.4A **SYSTCAL2.C**. The program implements the calibration simulation algorithm illustrated by fig.5.3. The following routines are found in the program which carry out the computational work for calibration:

GenTestData(): Routine to generate the 16 random target locations P(x,y) and calculate the expected anglescan measurements {α1,α2,α3,α4}.

MatrixMult(): General Matrix multiplication routine, any dimension matrix.

MatrixInvert(): General Matrix inversion with error output if the matrix is singular. Uses Gauss elimination technique and back substitution.

MatrixAdd(): Adds two matrices (must be the same dimensions)

MatrixPrint(): Displays a matrix on the screen. (for debugging)

InitPreEstimate(): Pre-estimates the initial values of the calibration coefficients.

NLLSE(): Actual calibration routine. Non-Linear Least Squares equation 5.31

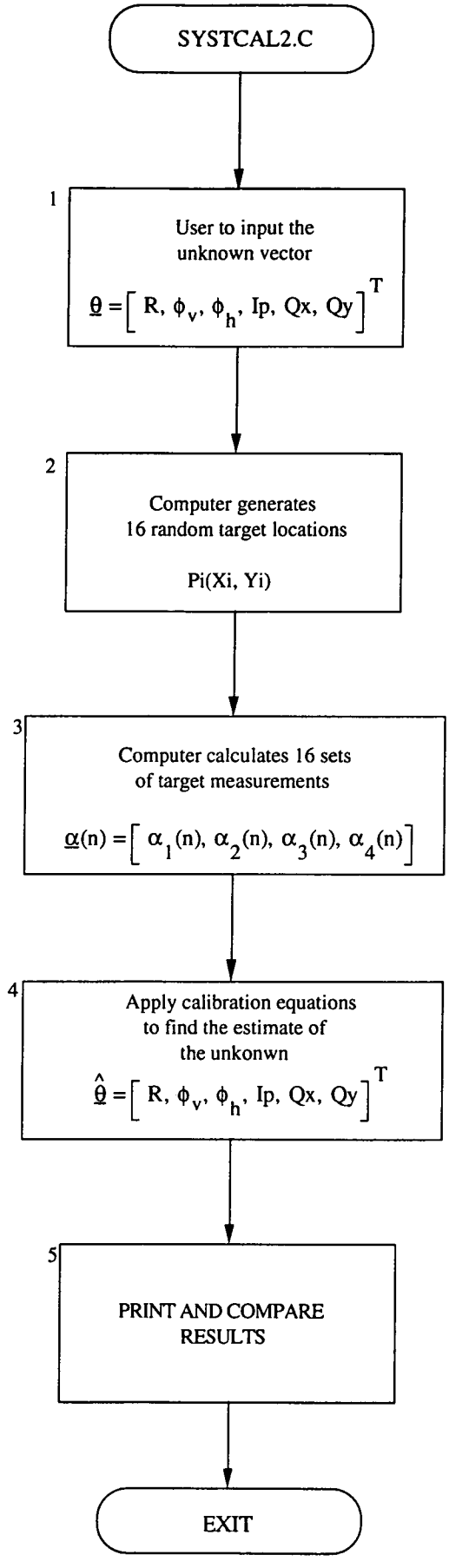


Fig.5.3 Calibration Flowchart Algorithm

(c) Simulation Results:

A typical simulation run using the program: "SYSTCAL2.C" is produced below, the user supplies the unknown input vector, the program will subsequently compute 16 random target locations, pre-estimate the unknowns, and finally apply the calibration algorithm.

User to supply the unknow input vector:

Instrument's Field of View(R) .....: 0.450 deg.
Vertical beam error(ϕv) .....: 2.500 deg.
Horizonal beam error(ϕh) .....: -1.500 deg.
Index pulse location(Ip) .....: -130.0 deg.
Offset to centre of field(Qx) .....: 0.800 deg.
Offset to centre of field(Qy) .....: -0.600 deg.

Computer generates 16 random target locations and measurements:

Note that the $\alpha_1,\alpha_2,\alpha_3,\alpha_4$ are not in degrees but in encoder output pulses in which 10000 pulses are produced per revolution.

X[0] =0.5798	Y[0] =-0.8232	α_1=303	α_2=2906	α_3=7002	α_4=9454
X[1] =0.7258	Y[1] =-0.8100	α_1=372	α_2=3450	α_3=6933	α_4=8911
X[2] =0.7351	Y[2] =-0.7273	α_1=690	α_2=3470	α_3=6615	α_4=8891
X[3] =0.8166	Y[3] =-0.7379	α_1=662	α_2=3760	α_3=6643	α_4=8600
X[4] =0.8901	Y[4] =-0.3975	α_1=1904	α_2=3969	α_3=5401	α_4=8391
X[5] =0.6987	Y[5] =-0.6251	α_1=1054	α_2=3323	α_3=6250	α_4=9037
X[6] =0.6240	Y[6] =-0.5108	α_1=1453	α_2=3026	α_3=5851	α_4=9334
X[7] =0.8289	Y[7] =-0.8063	α_1=398	α_2=3814	α_3=6907	α_4=8546
X[8] =0.6493	Y[8] =-0.4580	α_1=1648	α_2=3114	α_3=5656	α_4=9246
X[9] =0.8835	Y[9] =-0.4810	α_1=1586	α_2=3958	α_3=5719	α_4=8402
X[10]=0.9474	Y[10]=-0.3932	α_1=1927	α_2=4177	α_3=5377	α_4=8183
X[11]=0.6737	Y[11]=-0.6329	α_1=1024	α_2=3233	α_3=6281	α_4=9127
X[12]=1.0037	Y[12]=-0.4472	α_1=1724	α_2=4400	α_3=5581	α_4=7960
X[13]=0.9904	Y[13]=-0.4601	α_1=1674	α_2=4351	α_3=5631	α_4=8009
X[14]=0.7780	Y[14]=-0.5529	α_1=1317	α_2=3595	α_3=5987	α_4=8765
X[15]=0.8727	Y[15]=-0.5551	α_1=1318	α_2=3931	α_3=5987	α_4=8429

Computer generates initial pre-estimates for unknowns:

R=0.5000 ϕv=0.0000 ϕh=0.0000 Ip=-135.0000 Qx=0.000 Qy=0.000

Computer then solves for unknowns iteratively by least squares:

R=0.4487	ϕv=2.3443	ϕh=-1.2625	Ip=-130.5433	Qx=0.800	Qy=-0.600	Error=2551.941
R=0.4500	ϕv=2.5012	ϕh=-1.4991	Ip=-129.9984	Qx=0.800	Qy=-0.600	Error=9.541603
R=0.4500	ϕv=2.5000	ϕh=-1.5000	Ip=-130.0000	Qx=0.800	Qy=-0.600	Error=0.048369
R=0.4500	ϕv=2.5000	ϕh=-1.5000	Ip=-130.0000	Qx=0.800	Qy=-0.600	Error=0.000002
R=0.4500	ϕv=2.5000	ϕh=-1.5000	Ip=-130.0000	Qx=0.800	Qy=-0.600	Error=0.000001
R=0.4500	ϕv=2.5000	ϕh=-1.5000	Ip=-130.0000	Qx=0.800	Qy=-0.600	Error=0.000001
R=0.4500	ϕv=2.5000	ϕh=-1.5000	Ip=-130.0000	Qx=0.800	Qy=-0.600	Error=0.000002

The output of each iteration is shown above, the solution converges to the correct values on the 4th iteration. The program has been tested and found to be reliable at always converging to the correct solution. Note that equation (5.24) uses the rate of convergence $\{\lambda\}$ coefficient which has been set to $\lambda=1$ in this example.

The **Error** term in the last table is a measure of the RMS difference between the calculated values of P(x,y) and measured values:

$$\text{Error} = \sqrt{\frac{1}{4N}\left[(x-hx(\underline{\theta}))^2 + (y-hy(\underline{\theta}))^2 + (x-h\bar{x}(\underline{\theta}))^2 + (y-h\bar{y}(\underline{\theta}))^2 \right]} \qquad (5.32)$$

Equation (5.31) requires a the inversion of a matrix term. To invert a matrix, a subroutine was written called **MatrixInvert()** which uses Gauss elimination technique with scaled partial pivoting to generate an upper triangular matrix, this is then solved by back-substitution. If the matrix is singular, it returns an error condition.

Note that the unknown components Qx and Qy converge to the correct solution in one step, this is due to the fact that the equations $hx(\underline{\theta})$ and $hy(\underline{\theta})$ are a linear function of Qx and Qy, therefore the Taylor series approximation truncated to the first term is an exact equation which does not contain higher terms.

To pre-estimate the index pulse, the program starts scans form Ip=0 to 360 degrees in steps of 30 degrees calculating the error term at each point given by equation 5.32 The value of Ip which gives the smallest error is used as the initial estimated value of the index pulse.

[5.5] Current Calibration Implementation:

Before the developement work on this calibration scheme started, we had developed a much simpler calibration algorithm which is currently available on the anglescan computer. This simpler calibration scheme is found to work successfully and is described in this section. The more complex calibration algorithm could not be implemented on the current hardware platform because of memory (EPROM) limitation, the hardware only supports a maximum of 32K (RAM) and 32K (EPROM). Presently the program occupies the full EPROM capacity.
We have suggested that the microprocessor card be upgraded to either a 16 bit or 32 bit system as part of future development. This would be necessary in order to fully implement the target acquisition, tracking, and calibration algorithms.

Assumptions:

The simpler form of calibration assumes that the following parameters are known and therefore need not be determined from calibration:

(1) Field of view is R=0.5 degrees, set by the optical system
(2) Horizontal and vertical beam errors: $\theta_v=0$ and $\theta_h=0$.

Thus, the only unknowns remaining are:

(3) Index pulse Ip, range from -180 to +180 degrees.
(4) Centre of field Qx and Qy.

Note however that for target tracking purposes we do not need to know the centre of field with respect to some arbitary datum (Qx,Qy). Target tracking only relies on the relative position of the target with respect to the centre of field rather than absolute position to some datum. As such (4) above is not required.

Procedure:

The procedure involves placing the target mirror at two different points within the field of view of the instrument and taking measurements, the distance between the points must be known precisely (can be measured with a theodolite, or by using the calibration plate described earlier).

If the measurement points are denoted by: P1(x_1,y1) and P2(x_2,y_2), using equations (1.8A) and (1.8B):

Point P1(x_1,y_1): measurements: {α_1, α_2, α_3, α_4}

$$x_1 = \frac{R}{2}[\sin(\alpha_1 + Ip) + \sin(\alpha_3 + Ip)] \quad (5.33A)$$

$$y_1 = \frac{R}{2}[\cos(\alpha_2 + Ip) + \cos(\alpha_4 + Ip)] \quad (5.33B)$$

Point P2(x_2, y_2): measurements: {β_1, β_2, β_3, β_4}

$$x_2 = \frac{R}{2}[\sin(\beta_1 + Ip) + \sin(\beta_3 + Ip)] \quad (5.34A)$$

$$y_2 = \frac{R}{2}[\cos(\beta_2 + Ip) + \cos(\beta_4 + Ip)] \quad (5.34B)$$

The difference between x_2-x_1 and y_2-y_1 is known from measurements: thus:

$$\Delta x = x_2 - x_1 \quad (5.35A)$$

$$\Delta y = y_2 - y_1 \quad (5.35B)$$

substituting equations 5.33A and 5.34A into 5.35A, similarily, with equations 5.33B and 5.34B into equation 5.35B, and simplifying, we get the following relationship:

$$K_1\cos(Ip) + K_2\sin(Ip) = \frac{2.\Delta x}{R} \quad (5.36A)$$

$$K_3\cos(Ip) - K_4\sin(Ip) = \frac{2.\Delta y}{R} \quad (5.36B)$$

where the coefficients K_1, K_2, K_3, K_4 are:

$$K_1 = (\sin(\beta_1) - \sin(\alpha_1)) + (\sin(\beta_3) - \sin(\alpha_3)) \quad (5.37A)$$

$$K_2 = (\cos(\beta_1) - \cos(\alpha_1)) + (\cos(\beta_3) - \cos(\alpha_3)) \quad (5.37B)$$

$$K_3 = (\cos(\beta_2) - \cos(\alpha_2)) + (\cos(\beta_4) - \cos(\alpha_4)) \quad (5.37C)$$

$$K_4 = (\sin(\beta_2) - \sin(\alpha_2)) + (\sin(\beta_4) - \sin(\alpha_4)) \quad (5.37D)$$

solving equation 5.36A and 5.36B gives two possible values for the location of the index pulse:

$$Ip(x) = \text{atan}\left(\frac{K_2}{K_1}\right) \tag{5.38A}$$

$$Ip(y) = \text{atan}\left(-\frac{K_4}{K_3}\right) \tag{5.38B}$$

Where Ip(x) is due to the Δx measurement, and Ip(y) is due to the Δy measurement. Note that each equation 5.38A and 5.38B above may actually have two solutions which are 180 degrees apart, giving a total of 4 possible values of index pulse for each pair of measurements. To resolve this conflict, generally, several measurements are required before the index pulse can be determined accurately.
Note that a maximum of 4 different values can be obtained, although it is also possible that a pair of values are identical giving only 2 different outcomes.

The method described is currently implemented in C on the anglescan computer. The procedure utilizes the existing calibration plate placed at a known distance from the anglescan instrument, with the hole separation known precisely.
Placing the retro-reflective mirror at various points on the plate and performing calculations 5.38A and 5.38B a number of times allows the index pulse to be determined to an accuracy of 0.2 degrees This calibration technique has been used extensively during tracking or transient response measurements.
The calibration routine on the anglescan computer is called **DebugManCalib()** and is shown on page 2.40.

[5.6] Chapter Summary:

The more complex form of instrument calibration has not been implemented on the actual system. One of the problems with calibration is the handling of matrix algebra which is too slow on an 8 bit processor. Before implementing a full calibration algorithm, the microprocessor hardware should be upgrated to some better system such as a 68000 based processor or higher performance 16/32 bit system. Furthermore the current microprocessor is only capable of 32K EPROM memory, this is not sufficient to enable the full least squares calibration program to be loaded in memory.
The calibration simulation algorithm was written in C for an IBM-PC, similarily, the anglescan software system is also implemented in C (Z80). Porting the simulation software from the PC to the Z80 is relatively easy.

The simpler calibration procedure has being used extensively on the anglescan and found to work satisfactorily. We generally run a calibration on the instrument prior to testing the target tracking routines or conducting a step or PRBS response (for system identification reasons). Calibration is generally also required after the optical or mechanical system has being adjusted or modified.

Chapter-6

Discussion/Conclusion:

[6.1] General Applications:

The three most common applications of the anglescan tracking system include: (1) Surface contouring, (b) Tracking moving objects, and (c) General surveying.

(a) Surface Contouring:

The dual scan/track system is introduced in this section as part of future development work for applications requiring 3D surface contour mapping determination both for mining, archeology and industrial applications.
A recent paper describing this technique is given by G.Bayer [6.1], describing a system of two cameras for tracking moving vehicles such as tractors. It also shows a similar set up for target tracking as used in our section on controller design and performance evaluation (refer to Fig.4.1). A similar paper by M.Rioux [6.2] describes the same technique for mapping surfaces.

The dual scan/track system comprises of two anglescan instruments separated with a distance accurately measured. The first instrument consists of the anglescan system with a raster generation algorithm built into the program. This instrument scans across the X and Y planes. The second instrument is simply a passive detector which tracks the position of the light spot on the surface. The set up is illustrated in fig.6.1 below:

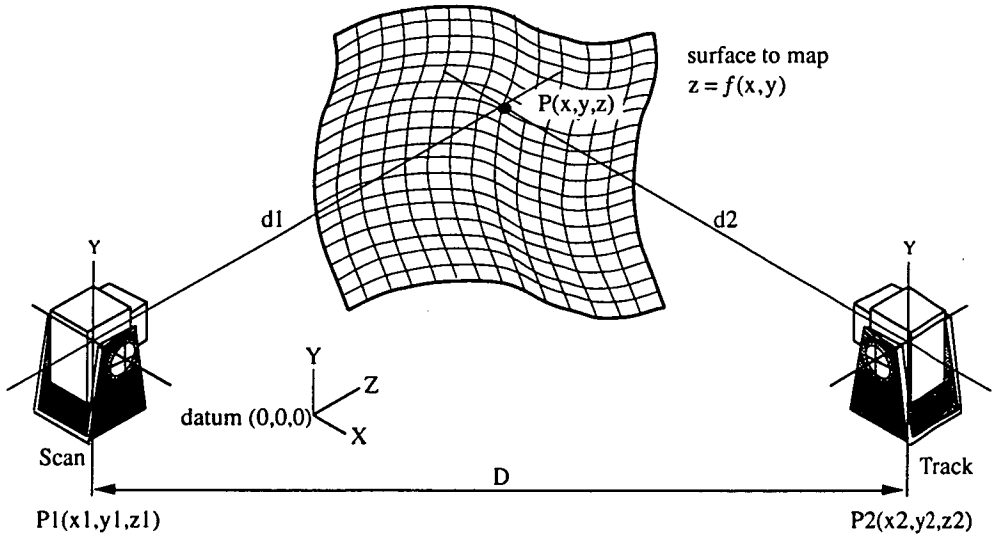


FIG.6.1 Scan and Track System Setup

Refering to fig.6.1, two anglescan instruments are separated by distance D. Each unit is positioned at P1(x1,y1,z1) and P2(x2,y2,z2) respectively with reference to some arbitary datum P(0,0,0).

The distance d1 and d2 can be measured in addition to the individual azimuth and elevation angles. This information is sufficient to enable the surface topography of an object to be mapped from geometry:
This application requires the use of multivariable control theory to model and investigate the performance of this set up.

(b) Tracking Moving Objects:

The anglescan tracking system is ideal for tracking moving vehicles such as tractors and bulldozers and likewise road paving vehicles where the surface contour should not deviate more than a specified amount. A paper by G. Bayer [6.1] discusses this particular application.
Another application in a paper by M.R.Driels [6.3] discusses the use of a tracking theodolite is in robot arm positioning calibration and accuracy determination. In this paper, a theodolite is used to track the motion of an end effector on an industrial robot to determine its positioning accuracy.
The instrument can also be used to track motions of large engineering structures such as bridges oscillating under wind gust conditions, or buildings.

(c) General Survey:

The anglescan system can be used for general field survey. Currenly the system has being tested and is capable of measuring azimuth and elevation angles to an accuracy of several seconds of arc to a distance of 700 meters.
One technique which can be used to improve the accuracy would be to use a raster scanning algorithm such that the target is repeatedly scanned across either the azimuth or elevation planes depending on the measurement required. With a large data set, better target positioning extrapolation techniques can be used such as least squares.

Alternatively, the joystick control can be used to manually point the instrument towards one target and then towards another target. During manual control the anglescan instrument reads displacement in whole integers from the optical encoder using the coarse positioning counter. However when the anglescan is on target it reads both whole integers and fractions using fine positioning counter described in section 2.5.

[6.2] Future Development Work:

Some of the more important aspects of the future development work include: (a) The testing of the linear quadratic controller and pole placement controller discussed in chapter 4, testing the calibration algorithm, (b) the implementation of a wide field search mode, (c) the implementation of different tracking modes, (d) development for potential commercilization of the system.

(a) Controller Implementation:

To implement the actual closed loop compensator on the anglescan computer, currently only simulation results were obtained. Actual controller implementation on the system are not available.
Neither one of the two controllers: Linear Quadratic and Pole placement have been implemented and tested on the actual system.

(b) Wide Field Mode:

The principle of wide field search mode is illustrated in fig.6.2 below:

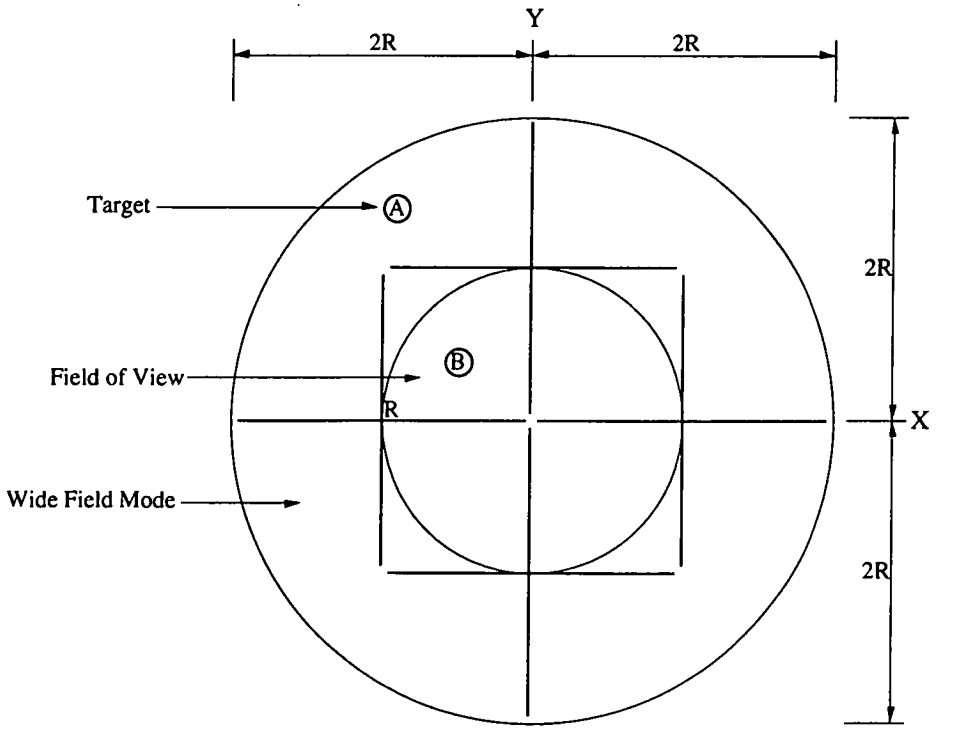


Fig.6.2 Wide Field Search Mode

Wide field search mode is used for target search and acquision, it accomplishes this task by utilizing the full horizontal and vertical beam span rather than simply the instrument's field of view. Fig.6.2 above illustrates the wide field mode, generally the innermost circle which is the instrument's field of view is used for continuous tracking, the target must be within this region, the field of view is 1.0 degrees (R=0.5 deg.) ex: target B.
The wide field is twice the diameter, represented by the outer circle and spans a field of view of 2.0 degrees, but cannot be used for tracking. Its only use is for target capture, ex: target A, furthermore, target determination in this mode produces some ambiguities about the actual location of the target and requires better interpolation algorithms.

(c) Tracking Modes:

A recent paper by: P.M.Salomon [1.4] describes various tracking modes used in a similar type of system.

search mode: search mode rotates the drives slowly about one axis until the target appears within the field of view, when detected automatic tracking mode is initiated. Open loop mode.

target acquisition and track: closes the tracking loop, and the instrument's field of view locks on the target. Closed loop mode.

flyback and sweep: initiated each time acquisition is lost, the tracker field of view is commanded to the flyback position (opposite to search position) and commanded to slowly sweep across the face of the tracker looking for the target. If target is not acquired on the sweep, the tracker is commanded into the search mode.

(d) Commercialization:

Some of the following suggestions were provided as future work to further simplify the instrument to a commercial unit:

[1] The use of an infrared solid state laser instead of the HeNe red laser. This will reduce the size considerably, and increase the mechanical robustness of the instrument. Additionally, the high voltage inverter is not required because the solid state laser can operate with 3V supplies.

[2] The hardware can be manufactured as 3 separate boards rather than the 6 currently used, thus: one microprocessor board, one return signal processor and target positioning counter board, and one motor driver board with the azimuth and elevation servo interface drivers and quadrature counters. Thus:

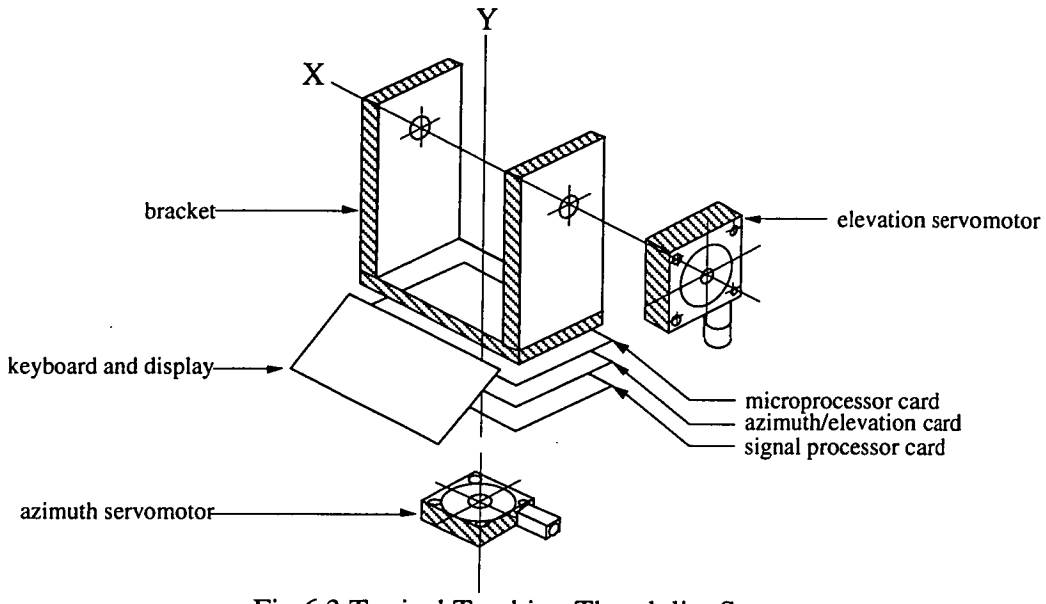


Fig.6.3 Typical Tracking Theodolite Setup

Fig.6.3 above illustrates the typical set up consisting of 3 main boards interconnected via a dedicated bus backplane and a front panel display/keyboard card. The 3 cards would include: (a) Either 16/32 bit off the shelf microprocessor card with floating point support, (b) A combined azimuth and elevation interface card identical to that previously described in section 2.5, (b) An analog signal processor card combined with a target measurement card, effectively 2.4 and 2.6 combined on one card.

(e) Distance Measurement:

Distance measurement can be integrated as part of the optical system, by intensity modulation of a solid state laser rather than the HeNe laser. Intensity modulation is simpler with a semiconductor laser compared with the gas laser by applying a modulating current to the semiconductor laser and using phase difference techniques previously described in section 1.5. Distance can be determined by switching different modulating frequencies beginning with the lowest frequency to obtain integer multiples, and progressively switching higher frequencies to obtain fine phase resolution. A typical EDM system is shown in Fig.6.4 below:

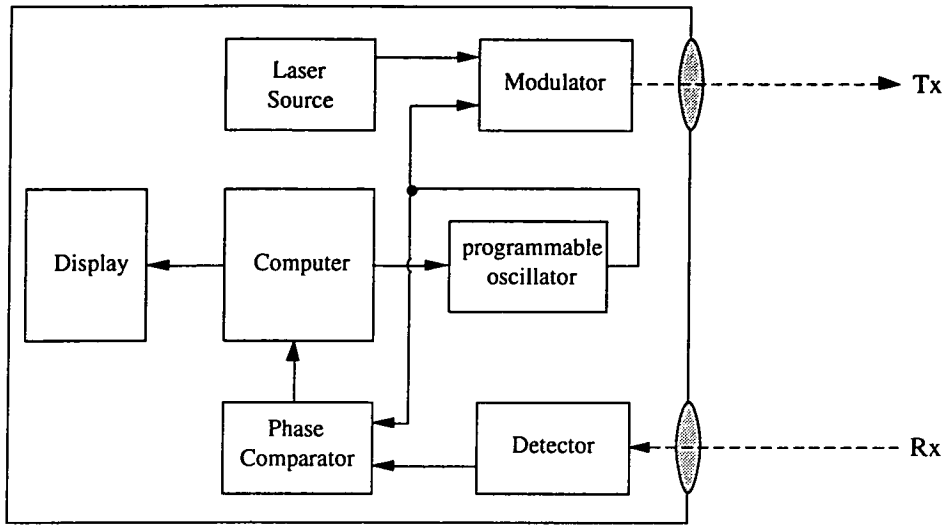


Fig.6.4 Typical EDM System

[6.3] Summary of Contributions:

We briefly summarize a list of work that has gone into the development of the anglescan tracking system.

(a) Hardware Design:

Chapter-2 describes the hardware which has being designed and built. The hardware includes the following cards designed and built on printed circuits:

(1) Two target measurement cards for measuring the location of the target,
(2) One azimuth PWM servo interface card,
(3) One elevation PWM servo interface card,
(4) One return laser analog signal processor card,
(5) Front panel keyboard and display card,
(6) One PID servo motor controller card (which was eventually replaced by the PWM controller).

Additionally, the construction of the power supply and chassis. Fig.2.2b in chapter-2 illustrates the entire assembly.

(b) Software Design:

The software system consists of about 100 pages of high level C source code, which was compiled with a HITECH-Z80C cross compiler on an IBM-PC and loaded into EPROM memory.
The software has being designed to enable simple hardware debugging and testing using commands from a serial terminal. Additionally, the software enables system identification tests to be conducted, transient response tests, calibration tests and closed loop auto-tracking tests for various controller implementations. A brief description of the software and the user interface is given in section 2.9 and in the appendix section 7.7A, B.

(c) Simulation and Modelling:

Simulation software was written for the purpose of evaluating several aspects of the anglescan system including complex calibration routines using least squares, system identification program designed to identify and model the two servo-systems, simulation programs to evaluate closed loop control system design including linear quadratic control and pole placement design.

The programs have being written in C for the IBM-PC and provide both graphical output to the screen (VGA mode) and output to an HP plotter (HP-GL type). A full listing of the software is included in the appendix, section 7.4.A, B, C, D, E.
The following programs were developed:

(a) Calibration simulation program (for chapter 5).
(b) Full system identification program (for chapter 3)
(c) Closed loop proportional control simulation (for chapter 4)
(d) Closed loop linear quadratic control simulation (for chapter 4)
(e) Closed loop pole placement design (for chapter 4).

[6.4] Conclusion:

The anglescan system was originally intended as a project to investigate the development of a fully automated surveying instrument. The instrument functions much like a theodolite but has extended applications. Because of the different nature of the measurent technique used (as compared to conventional theodolites), the system required a full custom design of the hardware (electronic system) and software system. Furthermore, custom design was necessary because of potential commercial applications.

Looking at the overall picture, the anglescan system basically consists of interdisciplinary work which is both theoretical, and practical in nature, combined to enable theoretical concepts to be directly applied into a practical working surveying instrument

Historical Perspectives:

We began looking into the system design from the early stages of the project, at that time only a working optical system was already constructed (see Sprent 1.11, 1.12) and a Z80 microprocessor board connected to the detector optics for some simple measurements. Because all the measurements were being conducted in software, this would tie up the microprocessor completely, it then became apparent that some form of measurement card which interfaced directly between the processor and the detector optics was necessary (refer to section 2.4) to releave the processor from timing loops. The function of the measurement card is to operate totally independent of the microprocessor and, at the end of the sampling period (one rotation of the cross) to latch the data on its output buffers, and make it available to the microprocessor when requested.

We later discovered after many measurements, that as the target was moved further away (or a smaller diameter target mirror), that the amplifier gain and threshold on the photodetector pre-amplifier needed adjusting in order to detect the return signal, this was the motivation behing the design of the analog signal processing card which was described in section 2.6. This card is fully programmable with filters to reduce noise and ambient stray light.

The next step was to design and build the servomotor interface and driver cards. The initial design consisted of a PID loop utilizing a dedicated Motion Controller IC. This design was found to be unsatisfactory (see 2.5) and we subsequently opted for an open loop PWM design. The microprocessor card effectively closes the loop, and an appropriate controller could then be implemented fully in software.

During the development of the hardware, software (interface) modules were also coded to debug and test each individual card.
Finally, we conducted PRBS measurements and transient response tests for system identification and controller design implementation purposes. We also developed several calibration strategies with the most powerful calibration which is described in chapter-5.
We also developed several controller simulation and design programs especially for the anglescan system model with the model of the measurement process also simulated in the control loop. This measurement process was found to have a destabilizing effect on the overall control system.

Acknowledgements:

I wish to acknowledge both my supervisor: Mr. G.The for the assistance in all the theoretical groundwork on system identification and control system design. I also wish to acknowledge Dr.A.Sprent for providing the financial resources and the opportunity to be involved in this project.

Chapter-7

Appendix:

[7.1A] General Notation:

Scalars: Scalars are denoted in lower case plain text, ex: x

Vectors: Vectors are denoted in lower case plain underlined text, ex: $\underline{\theta}$

Matrices: Matrices are denoted in upper case bold text, eg: $\mathbf{H}$

Subscripts:
- level of iteration, ex: θ_k
- variables belonging to same subset, ex: $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$
- sampling instance

Constants: Denoted in upper case plain text, ex: K

Estimates: Denoted by the original variable annotated by a hat symbol: eg: $\hat{x}$

Complex plane: Are represented in upper case thus H(z), H(s), H(jw), or polynomial functions A(q) with the delay operator q.

Time derivatives: Represented in dot notation: eg $\dot{e} = de/dt$

[7.1B] Mathematical Symbols:

α_1, α_2, α_3, α_4:	Four angular measurements obtained from the anglescan measurement card.
x, y :	Target position in azimuth and elevation from the centre of field of view, units of angular displacement are used.
X(z) :	Input reference signal to controller in s-plane, z-plane, and t-time domain
U(z):	Output signal from the controller to the plant (servosystem) in z-plane.
Y(z)	Output from the plant in z-plane
β, J	Coefficient of dynamic friction and load (rotational) inertia.
b1, b2, b3:	System identification coefficients (unkowns), numerator polynomial
a1, a2, a3:	System identification coefficients (unkowns), numerator polynomial. Can also denote angular measurements from the anglescan target measurement card.
R:	Field of view of the instrument's optics.
Kp, Ki, Kd:	PID filter coefficients: Proportional, integral, differential gain respectively
Ra, La:	Motor armature resistance R and inductance L respectively.
$\omega(z)$	Angular velocity on the output shaft in s-plane, z-plane and t-time domain.
$\theta c(z)$	Angular displacement on the output controller in z-domain
$\theta r(z)$	Angular displacement from the controller reference in z-domain
$B(q^{-1}),T(q^{-1}),S(q^{-1})$	Denominator polynomial representations in delay form. (For pole placement controller design only).
$A(q^{-1}), R(q^{-1})$	Numerator polynomial representations in delay form.
ek	Error term (scalar) in discrete time.

K_e, K_t:	Motor back EMF constant, and torque constant respectively.
$G_c(z)$	Controller transfer functions in z-plane.
$G(z)$	Servomotor model * measurement model ie: $G(z)=H(z).M(z)$
$H(z)$	Servomotor model transfer functions in z-plane
$M(z)$	Measurement process model dynamics
$G_m(z)$	Ideal (desired) closed loop transfer function used by the pole placement controller.
$R_{xy}(\tau,\tau)$:	Cross correlation function between the variables x(t) and y(t).
$R_{xx}(\tau,\tau)$:	Auto correlation function of the variable x(t).
$G_{xy}(j\omega)$:	Cross power spectral density of the function $R_{xy}(t,t)$
$G_{xx}(j\omega)$:	Auto power spectral density of the function $R_{xx}(t,t)$
$\underline{\theta}$:	-Unkown vector, $\underline{\theta}$ = [a1, a2, a3, b1, b2, b3] for system identification; -Unknown vector, $\underline{\theta}$ = [R, ϕ_v, ϕ_h, Ip, QX, QY] for system calibration.
i, j,k:	General indexing variables.
J_N:	Cost function or performance criterion.
tr[**H**]:	Trace of the matrix **H**.
V_{SAT}:	Saturation voltage of the PWM driver, set to 15Vdc.
V_c:	Coulomb friction, (deadband) in volts equivalent.
A, **B**	Matrices used in the representation of transfer function in state variable form.
$\underline{K}$	State feedback vector used in linear quadratic design
Q, **R**	Cost function weighting matrices for linear quadratic design.
Krl	Root locus gain by the servomotor dynamics.

[7.1C] Keywords and Definitions:

Some keywords encountered during the writing of this thesis:

Proper: A transfer function is proper if the degree of numerator polynomial >= degree of the denominator polynomial. deg{N(s)} >= deg{D(s)}.

Strictly Proper: A transfer function is proper if the degree of numerator polynomial > degree of the denominator polynomial. deg{N(s)} > deg{D(s)}.

Improper: A transfer function is said to be proper if the degree of the numerator polynomial < degree of the denominator polynomial. deg{N(s)} < deg{D(s)}.

Coprime: When the two polynomials N(s) and D(s) have no common factors.

Non minimum phase: The existance of a zero in the Right Hand s-plane, or outside the unit circle in the z-plane. Causes the transient step response to go negative at the beginning.

Plant leakage: All forward paths from the input x(t) to the output y(t) pass through the plant.

Monic: A polynomial is monic if the highest term coefficient is =1: ie:
$A(z) = zn + a1.zn\text{-}1 + ... + an\text{-}1.z + an$

Stability robustness: Closed loop system remains stable in spite of plant variations.

Performance robustness: The closed loop response remains acceptable in spite of plant variations.

[7.2A] Further Reading:

All references have being grouped into chapters headings, some references are not directly refered to in the thesis, however make interesting reading:

Chapter 1:

[1.1] G. Ammon, S. Russel,
"Application of Lasers, A Laser Tracking and Ranging System"
Applied Optics Vol. 9 No. 10 Oct. 1970 pp 2256-2260.

[1.2] Y. S. Lim, M. Subramanian,
"Improved Image Position Sensor For High Resolution Optical Tracking"
Applied Optics Vol. 11 No. 4 Apr. 1972 pp890-894.

[1.3] C.R. Cooke,
"Automatic Laser Tracking and Ranging System"
Applied Optics Vol. 11 No. 2 Feb. 1972 pp277-284

[1.4] P. M. Salomon, W. E. Crawford,
"A High Accuracy Digital Star Tracker"
Proceedings of the Society of Photo-Optical Instrumentation Engineers
Vol. 4, Oct 1972 pp85-97.

[1.5] A. Chrzanowski, F. Ahmed, B. Kurz,
"New Laser Applications in Geodetic and Engineering Survey"
Applied Optics, Vol. 11 No. 2 Feb. 1979 pp319-326

[1.6] T. J. M. Kennie, G.Petrie,
Engineering Surveying Technology"
First Edition 1990. (Chapter 1)

[1.7] P. S. Maybeck, D. E. Mercier,
"A Target Tracker Using Spatially Distributed Infrared Measurements"
IEEE Trans. on Automatic Control, Vol AC25 No. 2 Apr. 1980. pp222-225

[1.8] F. Deumlich,
"Surveying Instrumentation"
1972.

[1.9] S. H. Laurila,
"Electronic Surveying in Practice"
First Edition 1983

[1.10] D. M. Zipoy,
"Star Tracker for the Infrared"
Applied Optics, Vol. 25 No. 16 Aug. 1986 pp2845-2849.

[1.11] A. Sprent,
"Anglescan Tracking System"
Symposium on Surveillance and Monitoring Surveys,
University of Melbourne Nov. 1989 pp55-62.

[1.12] A. Sprent,
"EDM-Anglescan, An Automatic Earth Movement Monitoring System"
Symposium on Surveillance and Monitoring Surveys,
University of Melbourne Nov. 1986

[1.13] V. Dirita,
"A Laser Based Tracking Control System"
IREE Proceedings 1991 pp606-609.

[1.14] J. Ma, H. Sun, S. Wang, D. Yan,
"Effects of Atmospheric Turbulence on Photodetector Arrays"
Applied Optics Vol. 28, June 1989 pp2123-2126.

[1.15] L. G. Kazovsky,
"Theory of Tracking Accuracy of Laser Systems"
Optical Engineering, Vol. 22, No. 3, May-June 1983, pp339-347.

[1.16] L.Sher
"Tracking Systems Requirements for Atmospheric Steering Compensation"
Applied Optics Vol.14, No.11 Nov.1985

[1.17] F.E.Hoge
"Integrated Laser/Radar Satellite Ranging and Tracking System"
Applied Optics, Vol.13, No.3 October 1974

Chapter-2:

[2.1] P. Nandam, P. C. Sen
"A Comparative Study of PI and IP Controllers for DC Motor Drives"
International Journal of Control, Vol.44 No.1 pp283-297.

[2.2] A. F. Arbel
"Analog Signal Processing and Instrumentation"
1980, pp183-198

[2.3] J. Millman,
"Microelectronics Digital and Analog Circuits and Systems"
1979, pp595-596

[2.4] C. K. Dubey,
"Power Semiconductor Controlled Drives"
1989, pp35-62

Chapter-3:

[3.1] C.S. Burrus, T. W. Parks,
"DFT/FFT and Convolution Algorithms, Theory and Implementations" 1985

[3.2] A. Peled, B. Liu,
"Digital Signal Processing, Theory, Design and Implementation" 1976

[3.3] P.E.Wellstead, S.P.Sanoff,
"Extended Self Tuning Algorithms"
International Journal of Control, Vo. 34, No. 3, pp 433-455, 1981

[3.4] R.G.J. Acquot,
"Modern Digital Control Systems"

[3.5] L. Ljung,
"System Identification, Theory for the User" 1978

[3.6] H.J. Nussbaumer,
"Fast Fourier Tranform and Convolution Algorithms"
2nd Edition, 1982

[3.7] P. Eykhoff,
"System Identification, Parameter and State Estimation" 1974

[3.8] K. J. Hunt,
"A Survey of Recursive Identification Algorithms"
Transactions on Instrumentation, Measurement and Control, Vol. 8, No. 5,
pp. 273-278, 1986

[3.9] R. Stanway, J.E. Mottershead,
"Identification of Combined Viscous and Coulomb Friction. A Numerical
Comparison of Least Squares Algorithms."
Transactions on Instrumentation Measurement and Control, Vol. 8, No.1,
pp. 9-16, 1986.

[3.10] C. Morgan,
"Low Order Modelling Method for Z-Domain Transfer Functions"
Transactions on Instrumentation Measurement and Control, Vol.9, No.3,
pp.165-168, 1987

[3.11] M. T. Tham, S. N. Mansoori,
"Covariance Resetting in Recursive Least Squares Estimation"
IEE International Control 88, pp.128-133

[3.12] P. Eykhoff,
"Identification and System Parameter Estimation, Where Do We Stand Now?"
Automatica Vol.26, No.1 pp.3-5, 1990

[3.13] J. P. Norton,
"An Introduction to Identification"

[3.14] K. J. Astom, P. Eykhoff
"System Identification, A Survey"
Automatica, Vol.7, 1971, pp.123-162

[3.15] L .A. Zadeh,
"From Circuit theory to System Theory"
Proc. IRE 50, pp.856-865, 1966

[3.16] P.Eykhoff, P.C.Parks
"Identification and System Paramter Estimation, where do we stand now"
Automatica Vol.26, No.1, pp3-5, 1990

Chapter-4:

[4.1] K.J. Astrom, B. Wittenmark,
"Self Tuning Controllers Based on Pole-Zero Placement"
IEE Proceedings, Vol. 127, No.3 May 1980 pp120-130

[4.2] T.F. Chwee,
"Indirect Adaptive Pole-Assignment Controllers with Direct Implementations"
Transactions on Instrument, Measurement and Control Vol.10, No.4 July-Sep 1988 pp170-176

[4.3] K.J. Astrom
"Application of the Robust and Adaptive Pole Placement Design Technique."
International Journal of Adaptive Control and Signal Processing Vol.3 1989, pp167-189.

[4.4] J. Billingsley
"On the Design of Position Control Systems"
IEE Proceedings Vol.138 No.4 July 1991 pp 331-336

[4.5] C.L.Philips, H.T.Nagle
"Digital Control System Analysis and Design"
1984, pp271-301

[4.6] K.J. Astrom, B.Whittenmark,
"Adaptive Control"
1989, pp499-512

[4.7] M.Staron
"Control Optimization of a Laser Automatic Tracking System"
Applied Optics Vol.11, No.2, February 1972

[4.8] C.Canudas, K.J.Astrom, K.Braun
"Adaptive Friction Compensation in DC Motor Drives"
Journal of Robotics and Automation, Vol.RA-3, No.6, December 1987, pp.681-727

[4.9] S.Yang, M.Tomizuka
"Adaptive Pulse Width Control for Precise Positioning Under the Influence of Stiction and Coulomb Friction"
Journal of Dynamic Systems, Measurement and Control Sep.1988, Vol.110 pp221-227.

[4.10] P.N.Nikiforuk, K.Tamura
"Design of a Disturbance Accomodating Adaptive Control System and its Application to a DC-Servo Motor System with Coulomb Friction"
Journal of Dynamic Systems, Measurement, and Control Dec.1988, Vol.110, pp.343-349

[4.11] S.A.K. Al-Assadi, L.A.M.Chalabi
"Optimal Gain for Proportional Integral Derivative Feedback"
IEEE Control System Magazine, Dec.1987, pp.16-19

[4.12] P.K.Nandam, P.C.Sen
"A Comparative Study of Proportional Integral and Integral Proportional Controllers for DC Motors"
International Journal of Control 1986, Vol.44, No.1, pp.283-297

[4.13] D.W.Clarke
"PID Algorithms and Their Computer Implementation"
Transactions Instrumentation Measurement and Control Vol.6, No.6, Dec.1984 pp.305-316.

[4.14] B.C.Kuo
"Digital Control System"
1980 pp.509-514

[4.15] C.C.Hang, K.J.Astrom, W.K.Ho
"Refinements of the Ziegler Nichols Tuning Formula"
IEE Proceedings -D Vol.138, No.2, March 1991, pp.111-118

[4.16] H.S.Cho, C.W.Lee, Y.J.Cho
"Optimal Control System Design for Solar Tracking Concentrators"
Transactions Instrumentation Measurement and Control, Vol.2, No.4, Dec.1980, pp.207-213.

[4.17] C.C.Hang, K.K.Sin
"A comparative Performance Study of PID Auto-Tuners"
IEEE Control Systems, Aug.1991, pp.41-47.

[4.18] S.G.Wu, W.L.Chen
"Analysis and PID Controller Design of PWM Systems"
Journal of Dynamic Systems, Measurement and Control, Vol.110, Dec.1988 pp.355-360.

[4.19] M.Athans,
"The Status of Optimal Control Theory and Applications to Deterministic Systems"
IEEE Transactions on Automatic Control, July.1966, pp-580-592

[4.20] R.V.Patel, N.Munro
"Multivariable System Theory and Design"
1982 pp.201-233

[4.21] R.O.Huges,
"Optimal Control of Sun Tracking Solar Concentrators"
Journal of Dynamic Systems, Measurement and Control, June.1989, Vol.101 pp.157-161.

[4.22] C.T.Chen,
"Introduction to the Linear Algebraic Method for Control System Design"
IEEE Control System Magazine, Oct.1987, pp.36-42.

[4.23] K.J.Astrom,
"Assessment of Achievable performance of Simple Feedback Loops"
International Journal of Adaptive Control and Signal Processing, Vol.5, 1991 pp.3-19.

Chapter-5:

[5.1] E. M. Mikhail, G. Gracie,
"Analysis and Adjustments of Survey Measurements"
Chapter-4, 1981

[5.2] H. W. Sorenson,
"Parameter Estimation, Principles and Problems"
1980.

[5.3] L. Jung,
"System Identification, Theory For The User"
pp115-121

Chapter 6:

[6.1] G. Bayer, P. Krzystek, W. Mohlenbrink
"Real Time Positioning of Moving Objects by Dynamic Target Tracking"
ISPRS J. Photogrammetry and Remote Sensing 1991 pp.147-160.

[6.2] M. Rioux
"Laser Range Finder Based on Synchronized Scanners"
Applied Optics Vol.23, No.21, November 1984

[6.3] M.R.Driels,
"Vision-Based Automatic Theodolite for Robot Calibration"
IEEE Transactions on Robotics and Automation, Vol.7, No.3, June 1991, pp.351-360.

[7.3A] Least Squares Estimation:

The solution to the least squares equation is derived. Consider the linear equation representing a set of N measurements:

$$\underline{Y} = \mathbf{H}.\underline{\theta} + \underline{e} \tag{7.3.1}$$

Where Y=output vector, **H**=past observations and input matrix, and e=residual (error) vector, and $\underline{\theta}$=unknown vector. We require to minimize the cost function denoted by J defined as: (dispense with the underline to denote vectors):

$$J_N = \sum_{i=1}^{N} e_i^2 \tag{7.3.2}$$

$$= e^T.e \tag{7.3.3}$$

$$= [Y - \mathbf{H}.\theta]^T [Y - \mathbf{H}.\theta] \tag{7.3.4}$$

$$= Y^TY - Y^T\mathbf{H}.\theta - \theta^T\mathbf{H}^TY + \theta^T\mathbf{H}^T\mathbf{H}.\theta \tag{7.3.5}$$

to minimize the function Jn, set the first derivative to zero: $\partial J/\partial\theta=0$

$$\frac{\partial J}{\partial \theta} = \frac{\partial}{\partial \theta}[Y^TY] - \frac{\partial}{\partial \theta}[Y^T\mathbf{H}.\theta] - \frac{\partial}{\partial \theta}[\theta^T\mathbf{H}^YY] + \frac{\partial}{\partial \theta}[\theta^T\mathbf{H}^T\mathbf{H}.\theta] \tag{7.3.7}$$

$$0 = 0 - Y^T\mathbf{H} - Y^T\mathbf{H} + \theta^T\mathbf{H}^T\mathbf{H} + \theta^T\mathbf{H}^T\mathbf{H} \tag{7.3.8}$$

solving for the unknown θ above by transposing all terms and taking θ on one side gives the estimate:

$$\hat{\underline{\theta}} = \left[\mathbf{H}^T\mathbf{H}\right]^{-1}\mathbf{H}^T\underline{Y} \tag{7.3.9}$$

The product within the square brackets requires a matrix inversion function, the size of the matrix is MxM where M=number of unkowns to be identified. The result is an estimation represented by the 'hat' annotation.

[7.4] Simulation Programs:

[7.4A] System Calibration Simulation Program (SYSTCAL2C):

This program is used by the calibration simulation algorithm previously developed in chapter-5. The program applies least squares and solves for the unknown calibration coefficients using Newton-Raphson formula recursively.

[7.4B] System Identification program(SYSTIDE.C):

This program implements 8 different identification algorithms, recursive least squares, extended recursive least squares, non recursive least squares, 6 parameter model least squares, 3 parameter model least squares, recursive least squares with input offset, extended recursive least squares with input offset, non-recursive least squares with input offset. The output is in tabulated form.

[7.4C] Proportional Control System Simulation program(SYSTPROP.C)

This program is used for control system design and simulation using proportional control, it also includes the measurement model of the anglescan system. The output is plotted on an HP xy plotter.

[7.4D] LQR Control System Simulation program(SYSTLQR.C)

This program implements the linear quadratic regulator design and simulation described in section 4.6. The simulation also includes the measurement process. The output response is plotted on an HP xy plotter.

[7.4E] PPD Control System Simulation program(SYSTPPD.C)

This program implements the pole placement design and simulation described in section 4.6. The simulation also includes the measurement process. The output response is plotted on an HP xy plotter.

[7.5] Circuit Diagrams:

[7.5A] Signal Processor

[7.5B] Target Measurement Card

[7.5C] Motor Driver Card PWM

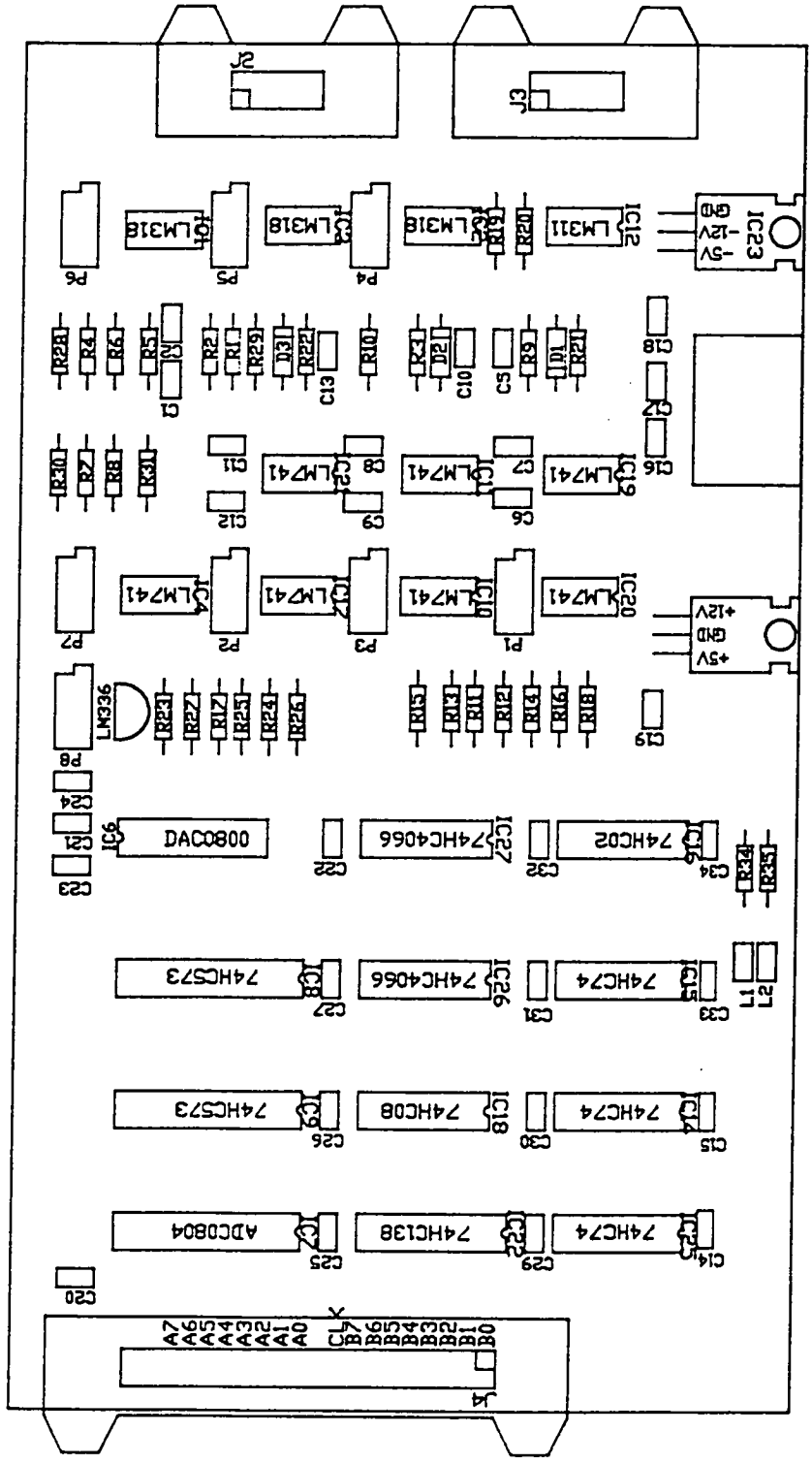
[7.5D] Front Panel Display Card

[7.5E] Motor Driver Card PID Controler

[7.5F] System Wiring Diagram

[7.5G] Joystick Manual Interface

[7.5A] Signal Processor Card: Circuit Diagram:



TO OPTO-DETECTOR

LASER IN

+15V

+15V

BUTTERWORTH 6dB/oLPF

1st STAGE BASELINE RESTORER
AMBIENT STRAYLIGHT

SIGNAL STRENGHT DETECTOR

2nd STAGE BASELINE DETECTOR

SUBTRACTOR

PROGRAMMABLE GAIN

GAIN SWITCHES

HIGH SPEED COMPARATOR

DIGITAL GLITCH FILTER

DIGITAL DELAY LINE

TO DUAL COUNTER

LASER OUT

IRMA A/D CLOCK

IRMA A/D CLOCK

CPU DATA BUS-A

TO CPU BUS

ADDRESS DECODE

POWER CONNECTOR

+15V

GND

-15V

POWER SUPPLY

A-GND

D-GND

NOTES:

[1] ALL RESISTORS 5% UNLESS OTHERWISE MARKED ON DRAWING.

[2] SEPARATE GROUND RAILS FOR A-GND, D-GND.

[3] BOARD SIZE 105x105mm ALIGNED WITH CPU HOLE CENTRES.

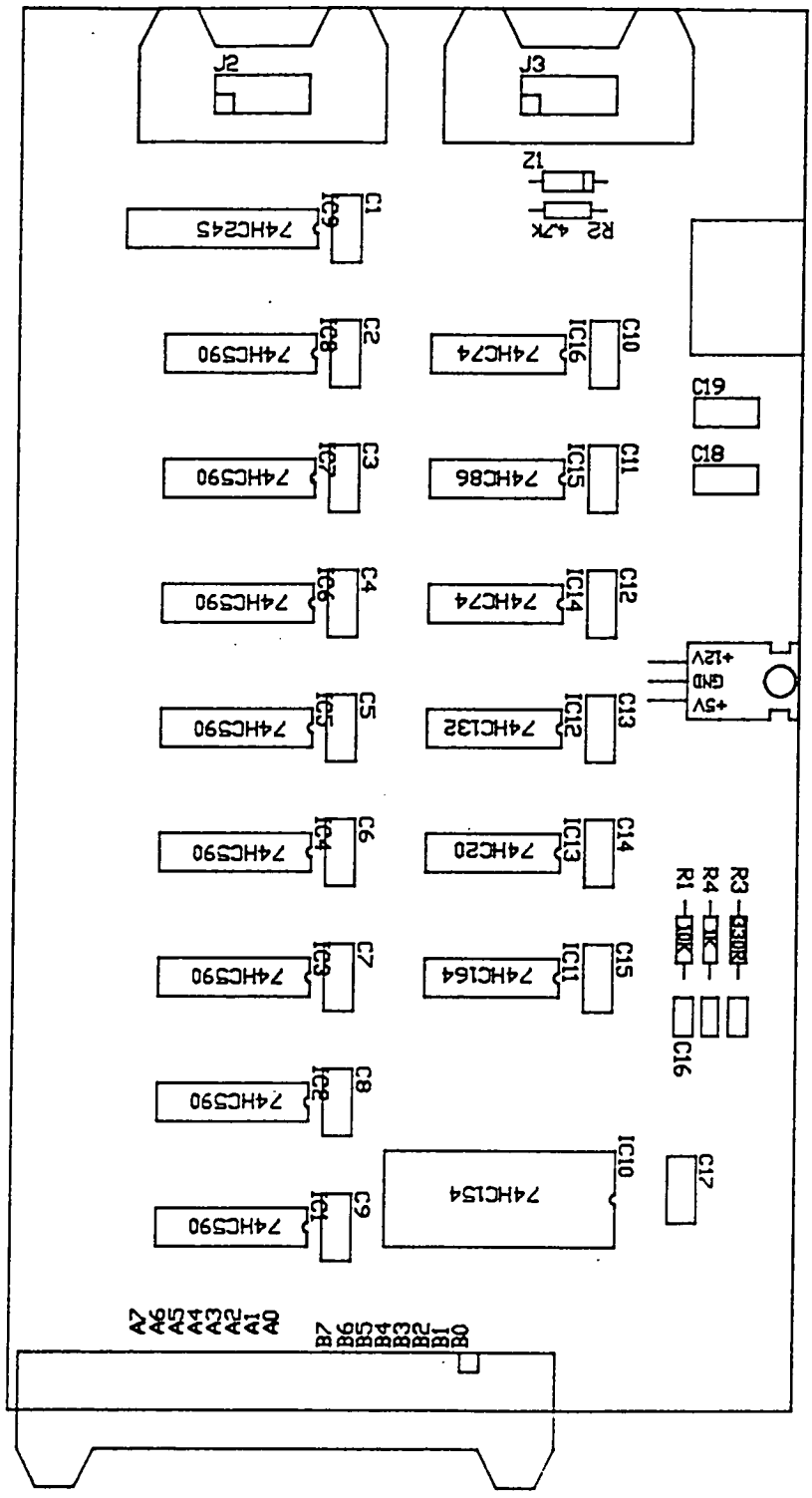
[4] ADDRESS DECODER: A/D (80):...: 74H
A/D (A0):...: 74H
GAIN:...: 74H
THRESHOLD:: 74H

TITLE
LASER RETURN SIGNAL PROCESSOR

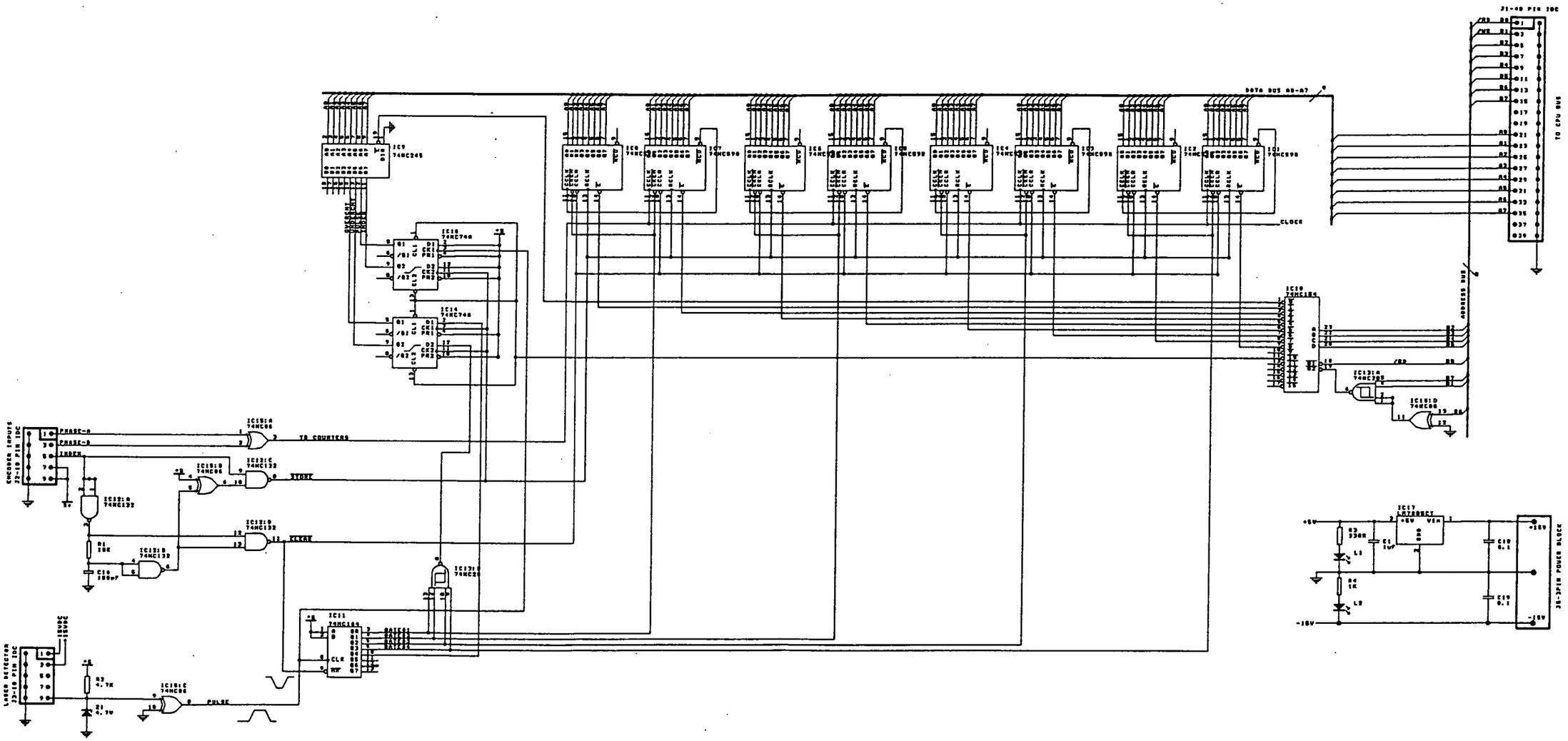
Size	Number	Revision
A1	ARSECH50.SB1	VER01

Sheet 1 of 1

[7.5B] Target Measurement Card: Circuit Diagram

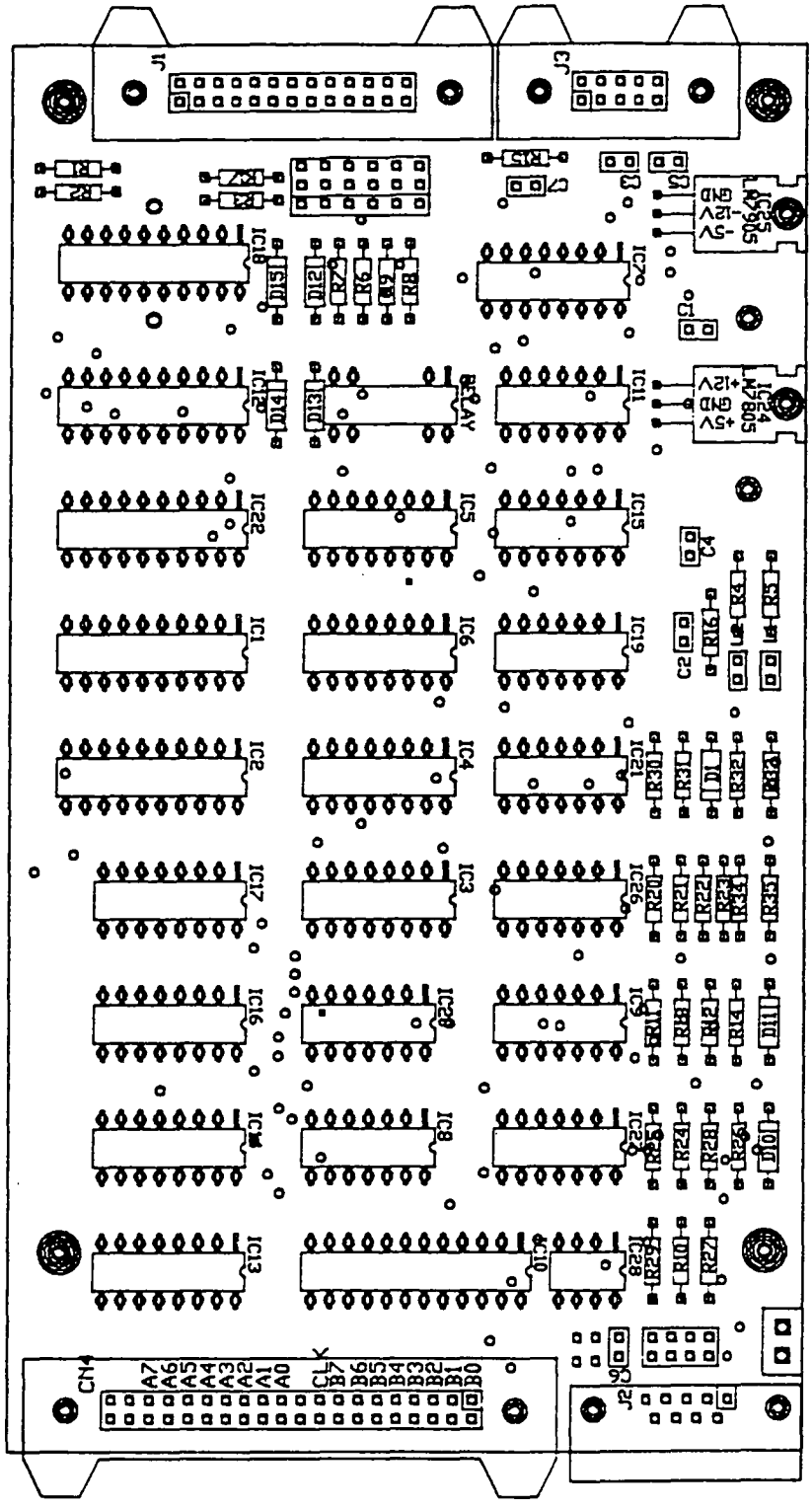


ANGSCN32 Top Overlay

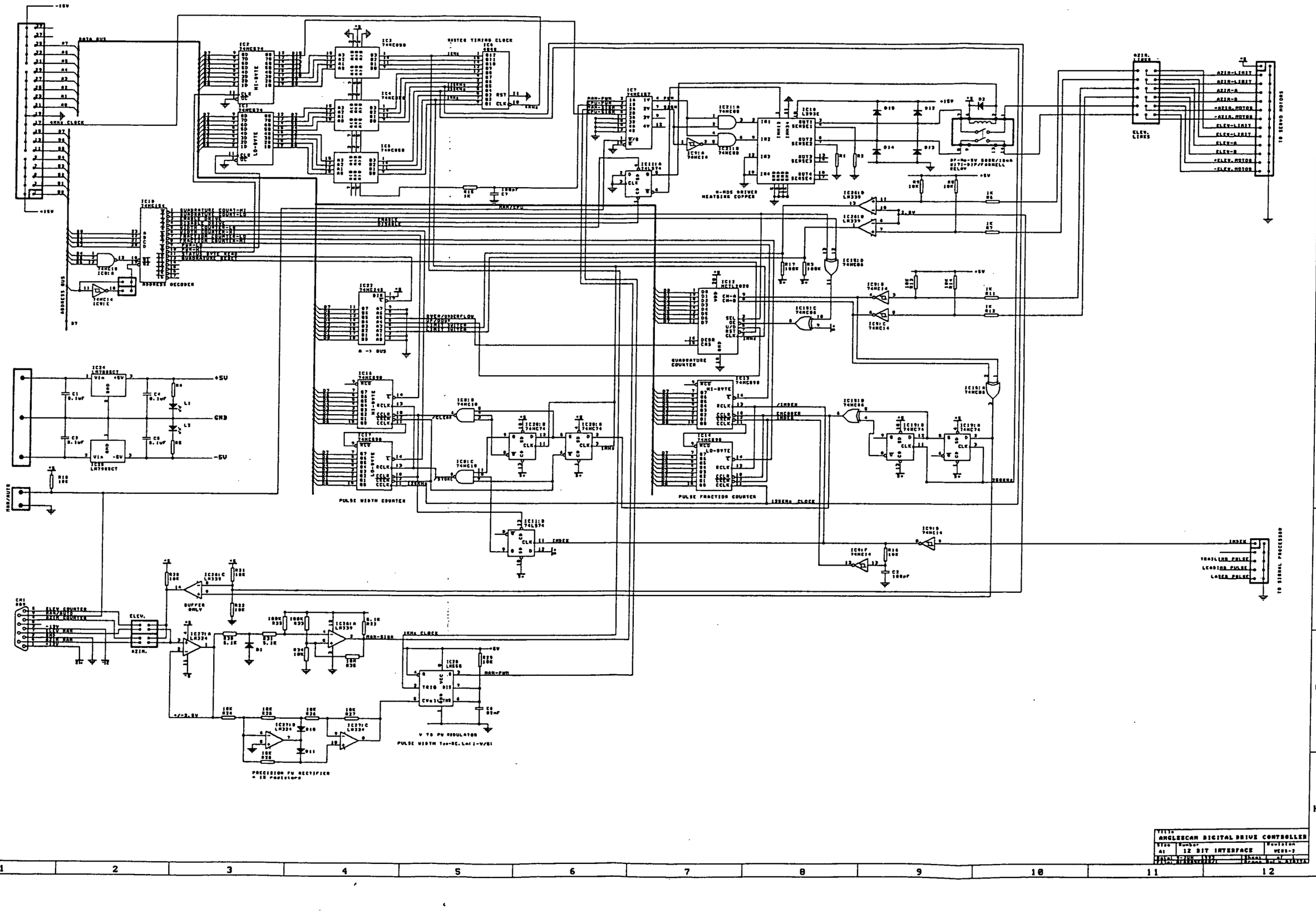


Title:	4 QUADRANT 16-BIT BINARY COUNTER
Size: A1	Number: 1
Revision:	VERS.9.2
Date: 6-FEB-1990	Time: 11:55:27
Sheet: 1 of 1	File: C:\PCB\VERS9\VERS9.SCH

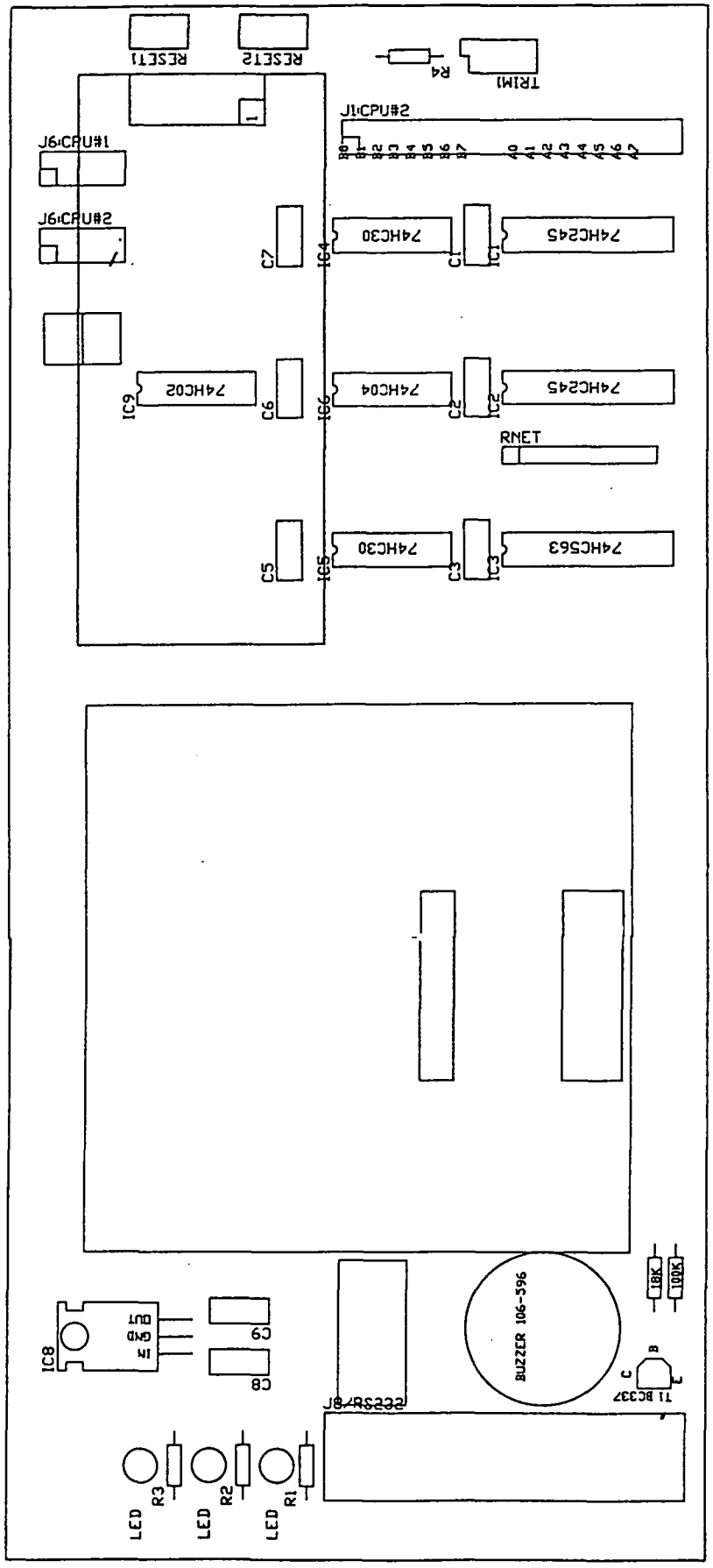
[7.5C] Motor Driver Card PWM: Circuit Diagram

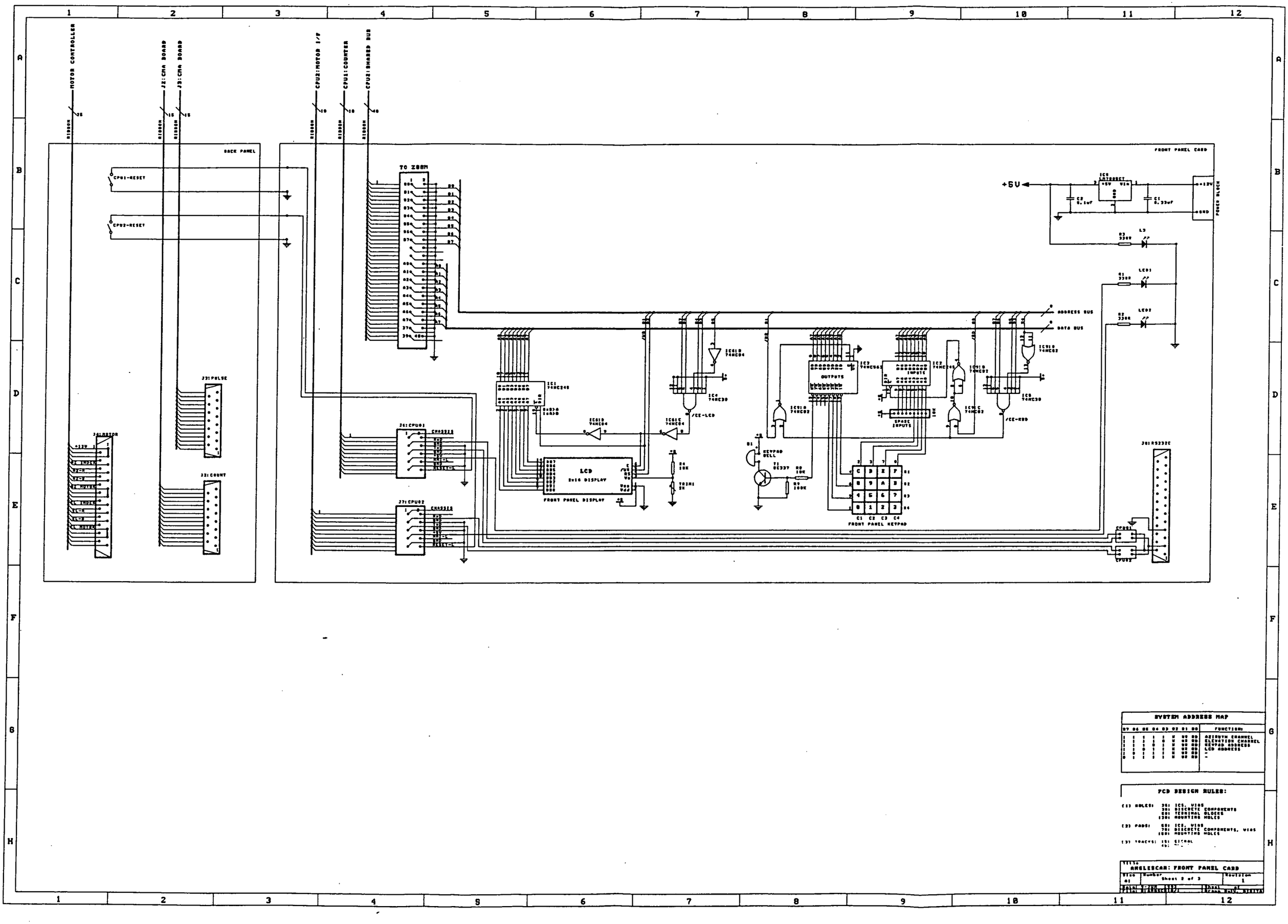


ENGSCN60 Check Plot


DATA BUS
4MHz CLOCK
MASTER TIMING CLOCK
ADDRESS DECODER
ADDRESS BUS
QUADRATURE COUNTER
PULSE WIDTH COUNTER
PULSE FRACTION COUNTER
H-MOS DRIVER HEATSINK COPPER
BUFFER ONLY
V TO PW MODULATOR
PRECISION PW RECTIFIER
TO SERVO MOTORS
TO SIGNAL PROCESSOR
INDEX
+5V
GND
-5V
ANGLESCAN DIGITAL DRIVE CONTROLLER
12 BIT INTERFACE

[7.5D] Front Panel Display Card: Circuit Diagram





SYSTEM ADDRESS MAP								
D7	D6	D5	D4	D3	D2	D1	D0	FUNCTION:
1	1	1	1	1	1	0	0	AZIMUTH CHANNEL
1	1	1	1	1	0	1	0	ELEVATION CHANNEL
1	1	1	1	1	1	0	0	KEYPAD ADDRESS
1	1	0	1	1	1	0	0	LCD ADDRESS
0	0	1	1	1	1	0	0	-
0	1	1	1	1	1	0	0	-

PCB DESIGN RULES:

(1) HOLES: 35: ICS, VIAS
30: DISCRETE COMPONENTS
50: TERMINAL BLOCKS
130: MOUNTING HOLES

(2) PADS: 65: ICS, VIAS
70: DISCRETE COMPONENTS, VIAS
100: MOUNTING HOLES

(3) TRACKS: 15: SIGNAL
45: "-"

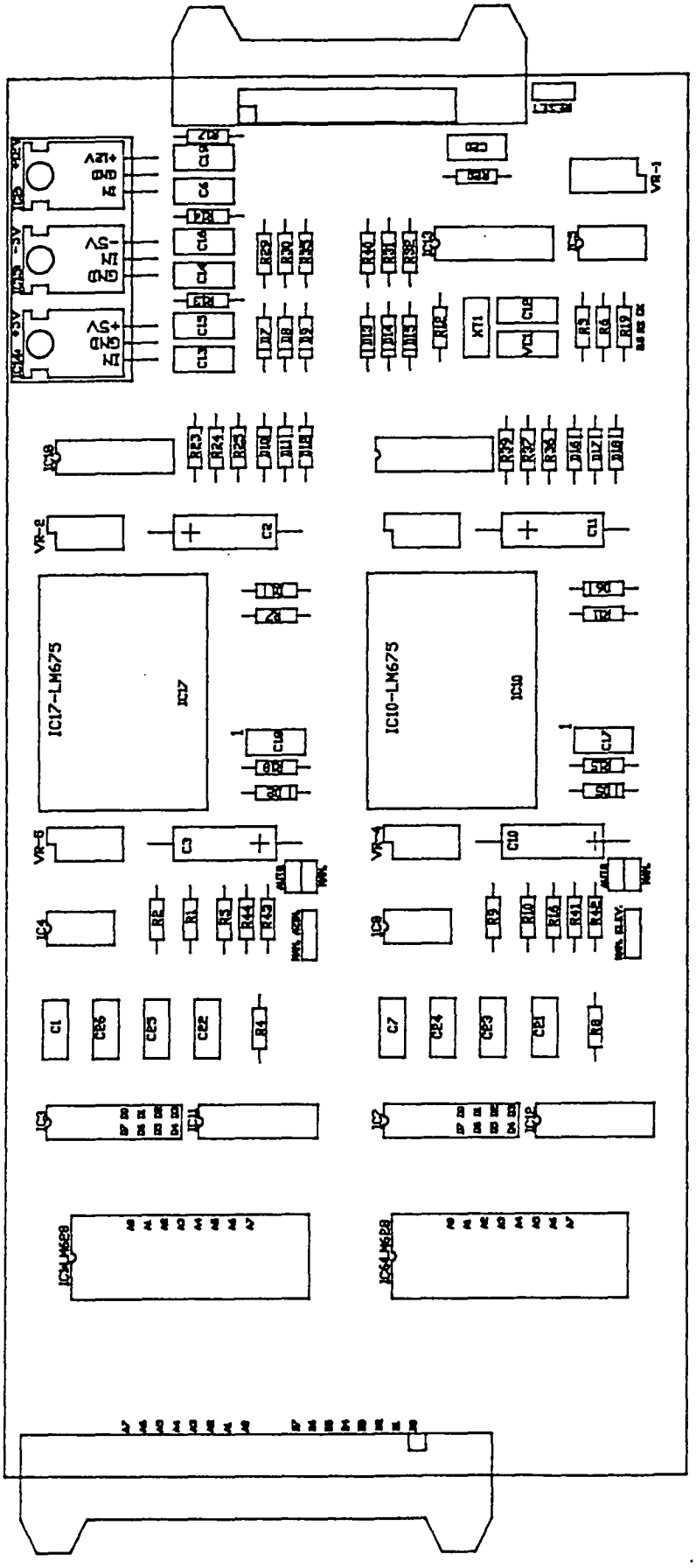
TITLE: ARGLESCAN: FRONT PANEL CARD

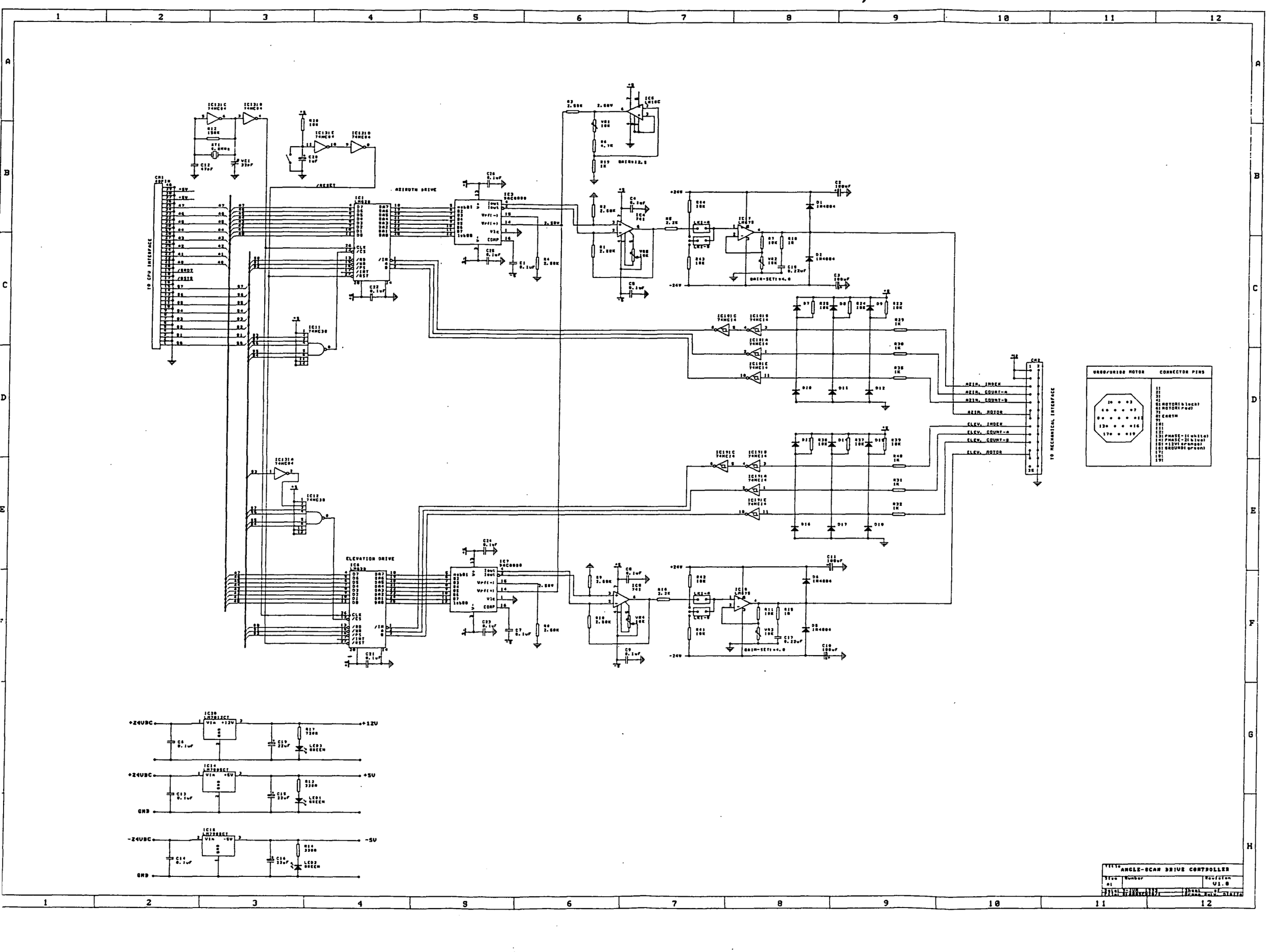
Size: A1 | Number: Sheet 2 of 3 | Revision: 1

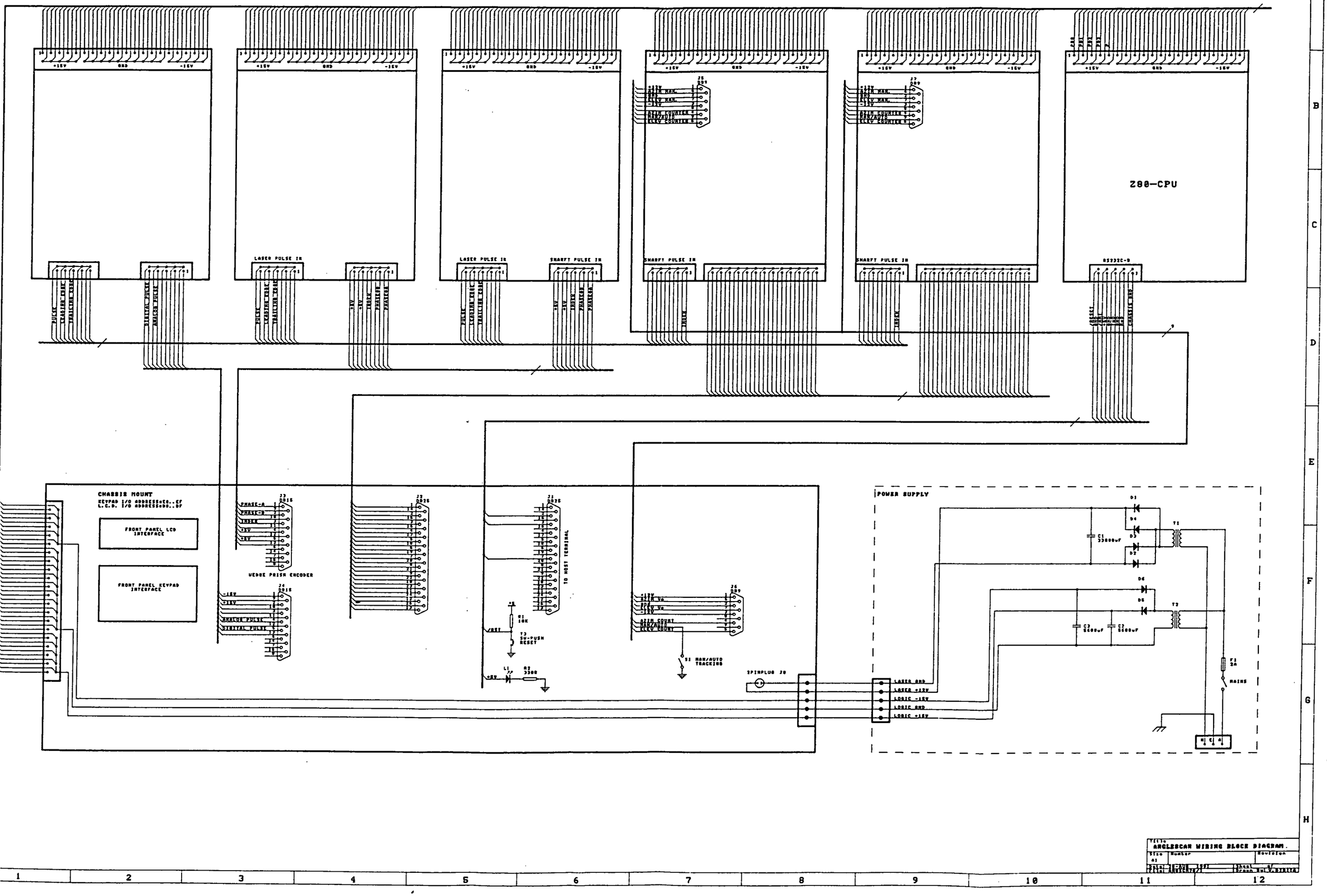
Date: 9-JAN-1989 | Sheet: 02

File: ARGLESCAN.SCH1V2 | Drawn By: STEVEN

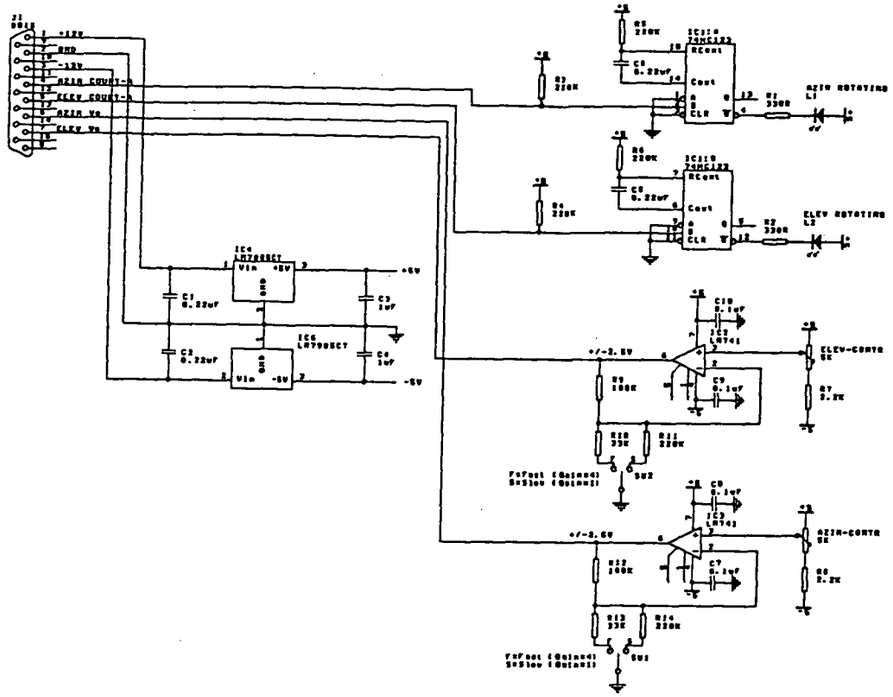
[7.5E] Motor Driver Card PID Controller: Circuit Diagram



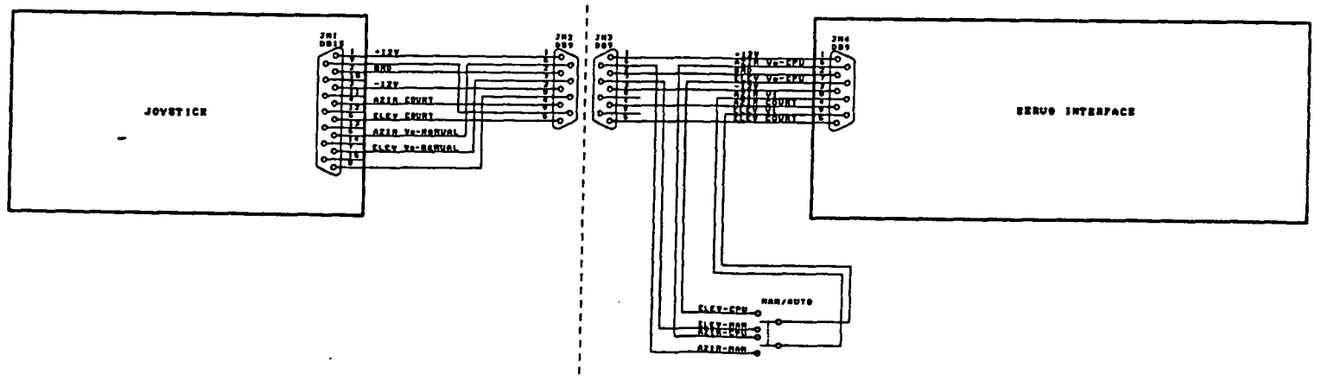

AZIMUTH DRIVE
ELEVATION DRIVE
/RESET
TO CPU INTERFACE
AZIM. INDEX
AZIM. COUNT-A
AZIM. COUNT-B
AZIM. MOTOR
ELEV. INDEX
ELEV. COUNT-A
ELEV. COUNT-B
ELEV. MOTOR
TO MECHANICAL INTERFACE
URSB/URISB MOTOR CONNECTOR PINS
+24VDC
-24VDC
+12V
+5V
-5V
GND
ANGLE-SCAN DRIVE CONTROLLER
Number
V1.0


Z80-CPU
LASER PULSE IN
SHARFT PULSE IN
RS232C-B
CHASSIS MOUNT
KEYPAD I/O ADDRESS=C0..CF
L.C.D. I/O ADDRESS=D0..DF
FRONT PANEL LCD INTERFACE
FRONT PANEL KEYPAD INTERFACE
WEDGE PRISM ENCODER
TO HOST TERMINAL
T3 SW-PUSH RESET
S1 MAN/AUTO TRACKING
3PINPLUG J8
POWER SUPPLY
LASER GND
LASER +12V
LOGIC -12V
LOGIC GND
LOGIC +12V
MAINS
ARGLESCAN WIRING BLOCK DIAGRAM.

JOYSTICK CONTROLLER



WIRING DIAGRAM



JOYSTICK MANUAL SERVO CONTROL

Size: A1 | Number: 1 | Revision: VERS-1

[7.7A] The User Interface:

Presently, the current software revision operating on the Anglescan tracking system has being developed to assist in debugging and to verify both the hardware and general operational features of the instrument. Consequently, when the Anglescan is connected to a host terminal (eg: serial RS232C, 9600 baud), the following menu appears on the screen: Refer to Fig.7.3 below: NOTE the / sign indicates that only one option is to be selected and compulsory:

Fig.7.3 Command Menu Structure

COMMAND	DESCRIPTION
H	List the Help Menu (this menu)
C	Calibrate the Instrument
D 1/2/b	Read Counter #1/#2/b=both
T	Fast Tracking Algorithm+Limit Filters
P a/e	PRBS Response on a=azimuth / e=elevation
I t=x.xx/g=xxx/a	g=xxx gain t=x.xx threshold a=ADC data
A p/r=xxxx/q/w/f	Azimuth p=xxxx(PWM) q=quadrature w=width f=fraction.
E p/r=xxxx/q/w/f	Elevation p=xxxx(PWM) q=quadrature w=width f=fraction.
S a/e=xxxx	Step Response a=azimuth e=elevation

COMMAND: H

Displays the Help menu on the screen, ie the above table 7.3. The help menu also appears on the screen upon a hardware reset or after a power up condition.

COMMAND: C

This command is used to calibrate the instrument manually. Calibration procedure requires the calibration plate to be placed at a precisely known distance from the anglescan instrument within the field of view. The retroreflective mirror is placed at some point on the calibration plate, the instrument makes the first measurement P1(x1,y1), the mirror is then placed at another point on the calibration plate and the instrument makes a further measurement P2(x2,y2). The operator is subsequently required to provide the angular distance Dx=x2-x1 and Dy=y2-y1, thereafter the instrument will perform the necessary calibration calculations and displays a number of results (because of ambiguity). The results include: FOV=field of view, and Ip=position of the index pulse with reference to the top dead centre. The calibration should be repeated in order to resolve the ambiguities and further increase accuracy. Calibration is described in section 5.8.

COMMAND: D 1/2/b

Command to read the four measurements off the position counter measurement board (refer to section 2.3). The rotary encoder provides resolution of N=10,000 counts/rotation, consequently the measurements are in encoder units and not in degrees.
Valid formats:

D 1 :reads counter #1 card.
D 2 :reads counter #2 card.
D b :reads both counters #1 card and #2 card.

COMMAND: T

Fast auto Tracking command. The anglescan tracking system software takes measurements every T=30 msec of target position, computes the azimuth and elevation angles, then applies a feedback control signal to the servo-motors according to the controller algorithm being tested.
The program prompts the user to supply: (1) Position of the index pulse (-180 to 180) (2) Compensator gain coefficients. If whilst tracking, the target is lost, the servomotors are disabled by default. (Target trajectory prediction algorithms are not used). To halt tracking, hit any key.

COMMAND: P a/e

PRBS test response. Used specifically for system identification. The controller applies a Pseudo-Random-Binary-Signal with a range from 0000 to 4095 (binary) to the servo-motors and measures the displacement. Up to 300 samples at T=30 msec per sample interval is used, giving a total test duration of 9 seconds. At the completion of the test, the instrument dumps all the measurements ie: input x(k) and output y(k) to the host terminal later to be used for system identification.
Valid input formats:

P a :PRBS test on the azimuth servo.
P e :PRBS test on the elevation servo.

COMMAND: I t=x.xx / g=xxx / a

This command is used to manually test and debug the analog signal processor card. During power up reset the software automatically presets the programmable gain amplifier and schmitt trigger comparator thresholds.

Valid Formats:

I t=x.xx	:set the threshold on the comparator in units of volts.
I g=xxx	:set the gain of the programmable amplifier stage.
I a	:read the ADC e: signal peak and baseline voltage

COMMAND: S a/e=xxxx

This command is used to conduct a simple transient step response on each of the two servosystems. It is used for system identification (chapter-3) as part of the model verification. The amplitude of the step can be specified in hundredth's of degrees.
Valid Formats:

S a=xxxx	:Azimuth servo step response xxxx=amplitude
S e=xxxx	:Elevation servo step response xxxx=amplitude

COMMAND: A p/r=xxxx/q/w/f

This command is used to test the azimuth interface card (refer to section 2.5). The p=xxxx applies a binary number to the Pulse Width Modulator with range 0 to 2^12, r=xxxx rotates the azimuth servo by xx.xx degrees, q=measures the main coarse rotary counter angular displacement which we refer to Ck in equation 2.3A, w=measures the width of the pulse which we refer to as N in equation 2.3A of the fine positioning counter, and finally f=measures the fraction of the pulse referred to as n in equation 2.3A of the fine positioning counter.
Valid Formats:

A p=xxxx	:apply binary value of bn=xxxx to PWM from 0 to 4095
A r=xxxx	:rotate the azimuth servo by xx.xx degrees
A q	:measure the main (coarse) rotary counter.
A w	:measure the fine positioning counter pulse width
A f	:measure the fine positioning counter pulse fraction

COMMAND: E p/r=xxxx/q/w/f

Identical to the command described above, except the command applies to the elevation interface card, and we refer to equation 2.3B instead of 2.3A above.

[7.7B] PIO Address Map:

The dedicated backplane bus consists of 2 ports namely PORT-A and PORT-B, the configuration is described below: Port-A is used as an 8 bit bidirectional data port, and Port-B is used as an 8 bit addressing port providing I/O to 256 port locations, the general format is illustrated below:

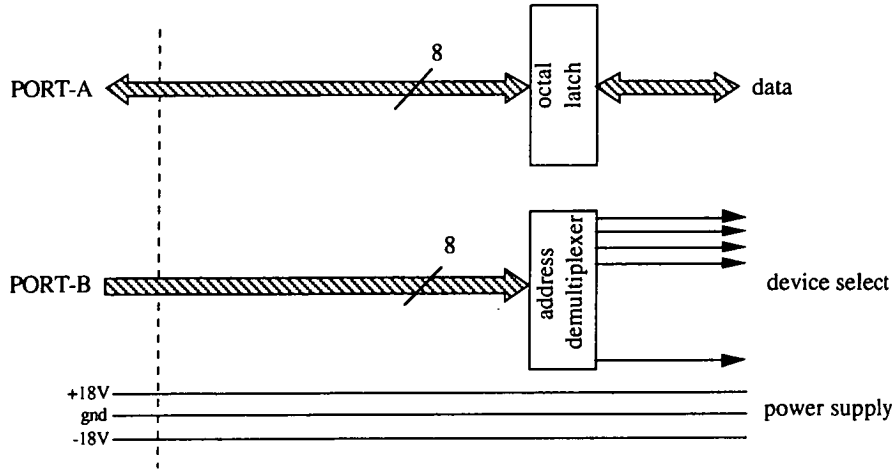


Fig.7.5 Bus Backplane Interconnections

Peripheral devices on port-B are address decoded, the following addresses are assigned to each of the peripheral interface boards which have being built:
On the azimuth and elevation card, a link is placed to enable the two cards to have different address maps.

serial RS232 ports:

0AH	RS232 channel-B communications data port
08H	RS232 channel-A communications data port
0BH	RS232 channel-B configuration control port
09H	RS232 channel-A configuration control port

Fig.7.6

PIO parallel ports:

04H	PIO 8 bit data register port-A
06H	PIO 8 bit data register port-B
05H	PIO configuration register port-A
07H	PIO configuration register port-B

Fig.7.7

azimuth motor interface card:

70H	Read azimuth coarse position quadrature rotary counter (high byte)
71H	Read azimuth coarse position quadrature rotary counter (low byte)
72H	Switch the azimuth motor relay on
73H	Switch the azimuth motor relay off
74H	Read azimuth fine positioning pulse width counter (low byte)
75H	Read azimuth fine positioning pulse width counter (high byte)
76H	Read azimuth fine positioning pulse fraction counter (low byte)
77H	Read azimuth fine positioning pulse fraction counter (high byte)
78H	Write azimuth PWM drive applied armature voltage (low byte)
79H	Write azimuth PWM drive applied armature voltage (high byte)
7AH	Read azimuth status byte
7BH	Reset the azimuth quadrature counter

Fig.7.8

elevation motor interface card:

F0H	Read elevation coarse position quadrature rotary counter (high byte)
F1H	Read elevation coarse position quadrature rotary counter (low byte)
F2H	Switch the elevation motor relay on
F3H	Switch the elevation motor relay off
F4H	Read elevation fine positioning pulse width counter (low byte)
F5H	Read elevation fine positioning pulse width counter (high byte)
F6H	Read elevation fine positioning pulse fraction counter (low byte)
F7H	Read elevation fine positioning pulse fraction counter (high byte)
F8H	Write elevation PWM drive applied armature voltage (low byte)
F9H	Write elevation PWM drive applied armature voltage (high byte)
FAH	Read elevation status byte
FBH	Reset the elevation quadrature counter

Fig.7.9

analog signal processor card:

37H	Select the progammable gain control latches
35H	Write to the programmable amplifer gain (lower 6 bits only)
3BH	Select the threshold control latches
39H	Write to the threshold control latches
33H	Select the A/D converter channel
32H	Read the A/D converter
31H	Write to the A/D converter reset.

Fig.7.13

front panel lcd display card:

DEH	LCD data read register
DDH	LCD data write register
DAH	LCD read control (status) register
D9H	LCD write to control register

Fig.7.10

position counter #1 card:

82H	Read counter status byte (overcount, undercount, no count)
86H	Read counter a1 counter (high byte)
8AH	Read counter a1 counter (low byte)
8EH	Read counter a2 counter (high byte)
92H	Read counter a2 counter (low byte)
96H	Read counter a3 counter (high byte)
9AH	Read counter a3 counter (low byte)
9EH	Read counter a4 counter (high byte)
A2H	Read counter a4 counter (low byte)
A6H	Reset the counters

Fig.7.11

position counter #2 card:

42H	Read counter status byte (overcount, undercount, no count)
46H	Read counter a1 counter (high byte)
4AH	Read counter a1 counter (low byte)
4EH	Read counter a2 counter (high byte)
52H	Read counter a2 counter (low byte)
56H	Read counter a3 counter (high byte)
5AH	Read counter a3 counter (low byte)
5EH	Read counter a4 counter (high byte)
62H	Read counter a4 counter (low byte)
66H	Reset the counters

Fig.7.12

[7.8] Photographs:

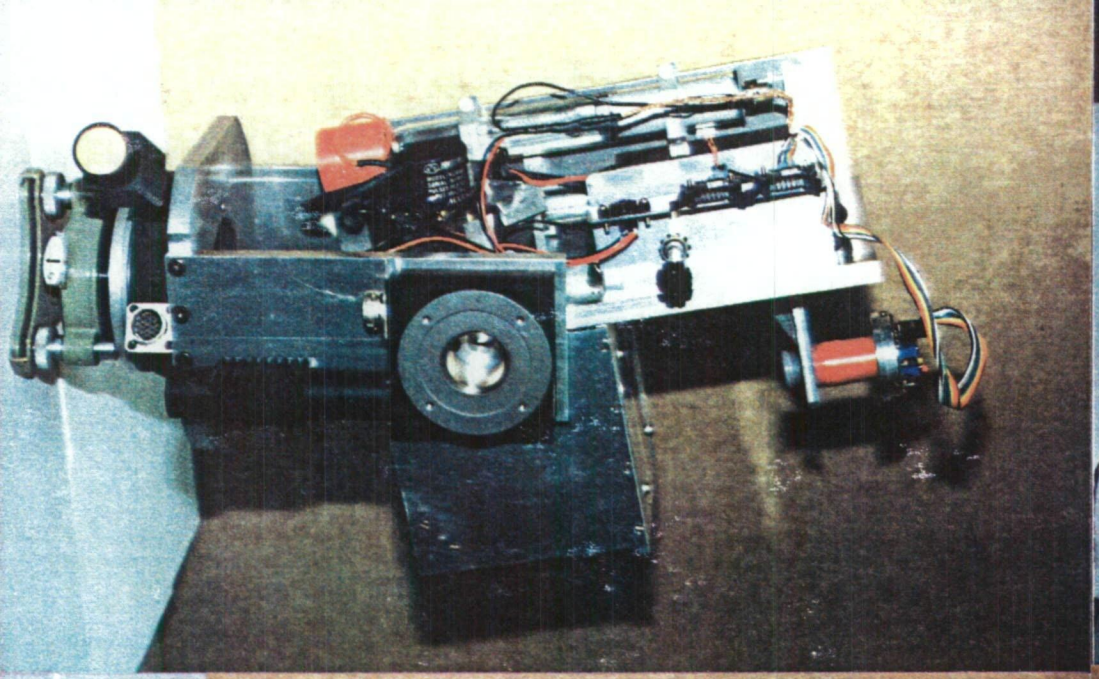
The photographs on the following page are numbered A,B,C,D beginning from the top left hand corner, top right hand corner, bottom left hand and right hand respectively.

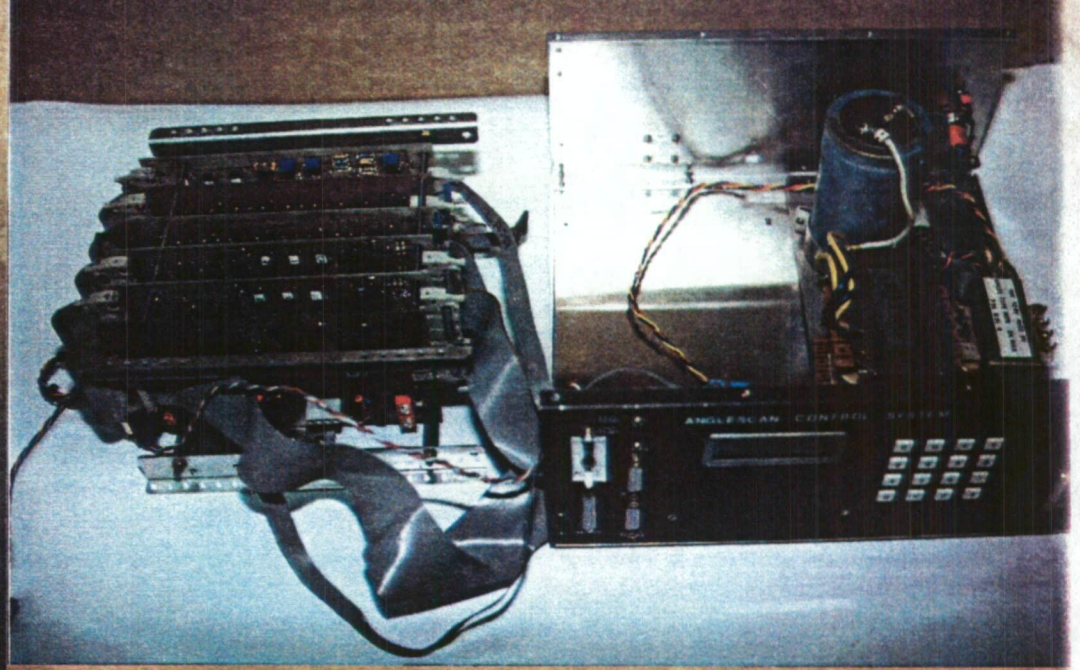
Photograph-A: Shows the complete electronics control unit comprising of an aluminium cabinet with the power supply mounted on the right, a front panel keypad and lcd display, the actual electronics is shown outside the aluminium box for clarity.

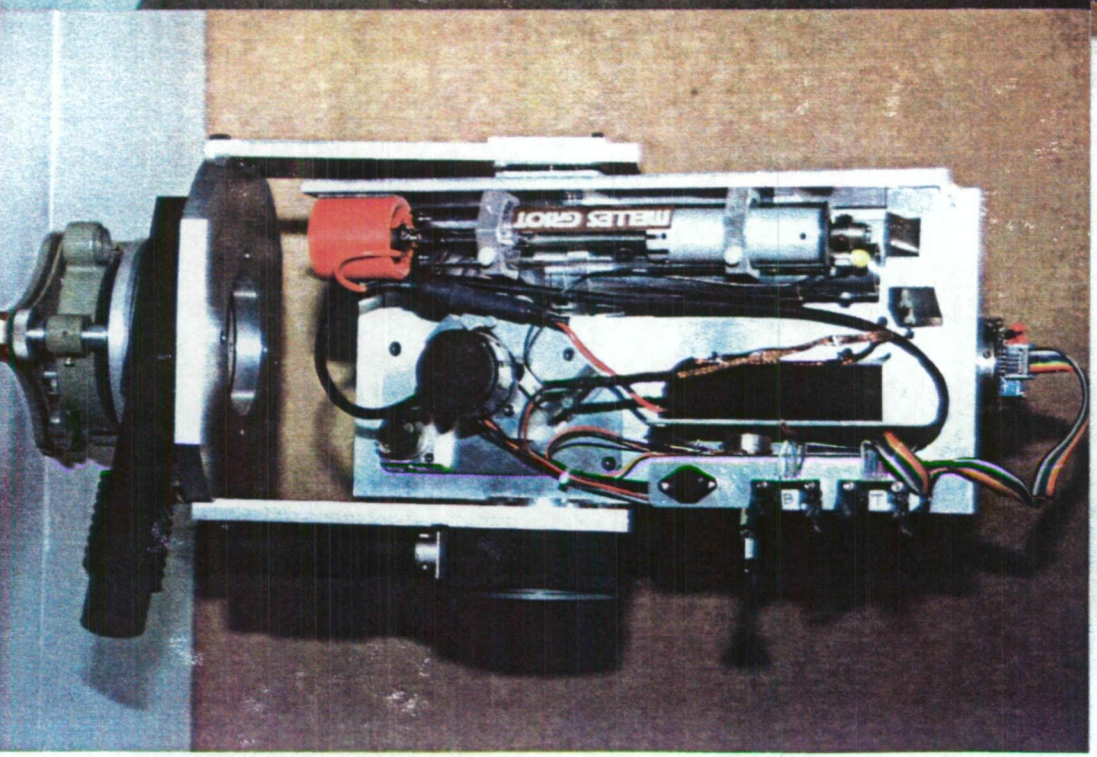
Photograph-B: Shows a more clear view of the electronics, the backplane bus actually consists of IDC (insulation displacement connectors) and ribbon cable. The first board (rightmost) is the CPU, the next two boards are the azimuth and elevation interface cards, the fouth and fith boards are the target measurement cards, and the last card (leftmost) is the signal processing card.

Photograph-C: Shows the complete optical unit generally mounted on a tripod. The servomotors (azimuth and elevation) are clearly visible. The optical photodetector is mounted at the top (inside the red cylinder).

Photograph-D: Shows the optical unit from the back. The laser is mounted vertically (leftmost). At the lower right hand the rotating wedge prism encoder is shown (black cylindrical casing) and just above it is the high voltage (7KV) laser power supply shown as a black rectangular block.




ANGLESCAN CONTROL SYSTEM


MEADES GIROT